

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 ComplexFlow Class Reference	9
5.1.1 Detailed Description	10
5.1.2 Constructor & Destructor Documentation	10
5.1.2.1 ComplexFlow()	10
5.1.3 Member Function Documentation	11
5.1.3.1 expression()	11
5.2 ExponentialFlow Class Reference	11
5.2.1 Detailed Description	12
5.2.2 Constructor & Destructor Documentation	12
5.2.2.1 ExponentialFlow()	12
5.2.3 Member Function Documentation	13
5.2.3.1 expression()	13
5.3 Flow Class Reference	13
5.3.1 Constructor & Destructor Documentation	14
5.3.1.1 ~Flow()	14
5.3.2 Member Function Documentation	14
5.3.2.1 expression()	14
5.3.2.2 getName()	14
5.3.2.3 getSystemBegin()	15
5.3.2.4 getSystemEnd()	15
5.3.2.5 getValue()	15
5.3.2.6 setName()	15
5.3.2.7 setSystemBegin()	15
5.3.2.8 setSystemEnd()	17
5.3.2.9 setValue()	17
5.4 FlowImplementation Class Reference	17
5.4.1 Detailed Description	19
5.4.2 Constructor & Destructor Documentation	19
5.4.2.1 ~FlowImplementation()	19
5.4.2.2 FlowImplementation()	19
5.4.3 Member Function Documentation	19
5.4.3.1 expression()	20

5.4.3.2 getName()	20
5.4.3.3 getSystemBegin()	20
5.4.3.4 getSystemEnd()	20
5.4.3.5 getValue()	20
5.4.3.6 setName()	20
5.4.3.7 setSystemBegin()	21
5.4.3.8 setSystemEnd()	21
5.4.3.9 setValue()	21
5.4.4 Member Data Documentation	22
5.4.4.1 name	22
5.4.4.2 systemBegin	22
5.4.4.3 systemEnd	22
5.4.4.4 value	22
5.5 LogisticalFlow Class Reference	23
5.5.1 Detailed Description	24
5.5.2 Constructor & Destructor Documentation	24
5.5.2.1 LogisticalFlow()	24
5.5.3 Member Function Documentation	24
5.5.3.1 expression()	24
5.6 Model Class Reference	25
5.6.1 Constructor & Destructor Documentation	25
5.6.1.1 ~Model()	26
5.6.2 Member Function Documentation	26
5.6.2.1 add() [1/2]	26
5.6.2.2 add() [2/2]	26
5.6.2.3 createFlow()	26
5.6.2.4 createModel()	27
5.6.2.5 createSystem()	27
5.6.2.6 endFlows()	27
5.6.2.7 endModels()	27
5.6.2.8 endSystems()	27
5.6.2.9 getFlowsIterator()	27
5.6.2.10 getModelsIterator()	28
5.6.2.11 getName()	28
5.6.2.12 getSystemsIterator()	28
5.6.2.13 getTime()	28
5.6.2.14 setName()	28
5.6.2.15 setTime()	29
5.6.2.16 simulate()	29
5.7 ModelImplementation Class Reference	29
5.7.1 Detailed Description	31
5.7.2 Constructor & Destructor Documentation	31

5.7.2.1 <code>~ModelImplementation()</code>	31
5.7.2.2 <code>ModelImplementation()</code>	31
5.7.3 Member Function Documentation	32
5.7.3.1 <code>add()</code> [1/2]	32
5.7.3.2 <code>add()</code> [2/2]	32
5.7.3.3 <code>createModel()</code>	32
5.7.3.4 <code>createSystem()</code>	33
5.7.3.5 <code>endFlows()</code>	33
5.7.3.6 <code>endModels()</code>	33
5.7.3.7 <code>endSystems()</code>	33
5.7.3.8 <code>getFlowsIterator()</code>	33
5.7.3.9 <code>getModelsIterator()</code>	33
5.7.3.10 <code>getName()</code>	34
5.7.3.11 <code>getSystemsIterator()</code>	34
5.7.3.12 <code>getTime()</code>	34
5.7.3.13 <code>setName()</code>	34
5.7.3.14 <code>setTime()</code>	34
5.7.3.15 <code>simulate()</code>	35
5.7.4 Member Data Documentation	35
5.7.4.1 <code>flows</code>	35
5.7.4.2 <code>models</code>	35
5.7.4.3 <code>name</code>	35
5.7.4.4 <code>systems</code>	36
5.7.4.5 <code>time</code>	36
5.8 System Class Reference	36
5.8.1 Constructor & Destructor Documentation	36
5.8.1.1 <code>~System()</code>	37
5.8.2 Member Function Documentation	37
5.8.2.1 <code>getName()</code>	37
5.8.2.2 <code>getValue()</code>	37
5.8.2.3 <code>setName()</code>	37
5.8.2.4 <code>setValue()</code>	37
5.9 SystemImplementation Class Reference	38
5.9.1 Detailed Description	39
5.9.2 Constructor & Destructor Documentation	39
5.9.2.1 <code>~SystemImplementation()</code>	39
5.9.2.2 <code>SystemImplementation()</code>	39
5.9.3 Member Function Documentation	40
5.9.3.1 <code>getName()</code>	40
5.9.3.2 <code>getValue()</code>	40
5.9.3.3 <code>setName()</code>	40
5.9.3.4 <code>setValue()</code>	40

5.9.4 Member Data Documentation	41
5.9.4.1 name	41
5.9.4.2 value	41
5.10 UnitTestFlow Class Reference	41
5.10.1 Detailed Description	42
5.10.2 Constructor & Destructor Documentation	42
5.10.2.1 UnitTestFlow()	42
5.10.3 Member Function Documentation	43
5.10.3.1 expression()	43
5.11 UnitTestFlow2 Class Reference	43
5.11.1 Detailed Description	44
5.11.2 Constructor & Destructor Documentation	44
5.11.2.1 UnitTestFlow2()	44
5.11.3 Member Function Documentation	45
5.11.3.1 expression()	45
6 File Documentation	47
6.1 cmake-build-debug/CMakeCache.txt File Reference	47
6.2 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c File Reference	47
6.2.1 Macro Definition Documentation	47
6.2.1.1 __has_include	48
6.2.1.2 ARCHITECTURE_ID	48
6.2.1.3 C_VERSION	48
6.2.1.4 COMPILER_ID	48
6.2.1.5 DEC	48
6.2.1.6 HEX	48
6.2.1.7 PLATFORM_ID	49
6.2.1.8 STRINGIFY	49
6.2.1.9 STRINGIFY_HELPER	49
6.2.2 Function Documentation	49
6.2.2.1 main()	49
6.2.3 Variable Documentation	49
6.2.3.1 info_arch	49
6.2.3.2 info_compiler	49
6.2.3.3 info_language_extensions_default	50
6.2.3.4 info_language_standard_default	50
6.2.3.5 info_platform	50
6.3 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference	50
6.3.1 Macro Definition Documentation	51
6.3.1.1 __has_include	51
6.3.1.2 ARCHITECTURE_ID	51
6.3.1.3 COMPILER_ID	51

6.3.1.4 CXX_STD	51
6.3.1.5 DEC	51
6.3.1.6 HEX	52
6.3.1.7 PLATFORM_ID	52
6.3.1.8 STRINGIFY	52
6.3.1.9 STRINGIFY_HELPER	52
6.3.2 Function Documentation	52
6.3.2.1 main()	52
6.3.3 Variable Documentation	52
6.3.3.1 info_arch	53
6.3.3.2 info_compiler	53
6.3.3.3 info_language_extensions_default	53
6.3.3.4 info_language_standard_default	53
6.3.3.5 info_platform	53
6.4 cmake-build-debug/CMakeFiles/clion-environment.txt File Reference	54
6.5 cmake-build-debug/CMakeFiles/clion-log.txt File Reference	54
6.6 cmake-build-debug/CMakeFiles/TargetDirectories.txt File Reference	54
6.7 CMakeLists.txt File Reference	54
6.8 README.md File Reference	54
6.9 src/lib/Flow.h File Reference	54
6.10 Flow.h	55
6.11 src/lib/FlowImplementation.cpp File Reference	56
6.12 src/lib/FlowImplementation.h File Reference	57
6.13 FlowImplementation.h	57
6.14 src/lib/Model.h File Reference	58
6.15 Model.h	59
6.16 src/lib/ModelImplementation.cpp File Reference	60
6.17 src/lib/ModelImplementation.h File Reference	62
6.18 ModelImplementation.h	63
6.19 src/lib/System.h File Reference	64
6.20 System.h	64
6.21 src/lib/SystemImplementation.cpp File Reference	65
6.22 src/lib/SystemImplementation.h File Reference	65
6.23 SystemImplementation.h	66
6.24 src/main.cpp File Reference	67
6.24.1 Function Documentation	67
6.24.1.1 main()	67
6.25 test/functional/main.cpp File Reference	67
6.25.1 Function Documentation	68
6.25.1.1 main()	68
6.26 test/unit/main.cpp File Reference	68
6.26.1 Function Documentation	69

6.26.1.1 main()	69
6.27 test/functional/FunctionalTests.cpp File Reference	70
6.27.1 Function Documentation	70
6.27.1.1 ComplexTest()	70
6.27.1.2 ExponencialTest()	70
6.27.1.3 LogisticalTest()	71
6.28 test/functional/FunctionalTests.h File Reference	71
6.28.1 Function Documentation	72
6.28.1.1 ComplexTest()	72
6.28.1.2 ExponencialTest()	72
6.28.1.3 LogisticalTest()	72
6.29 FunctionalTests.h	72
6.30 test/unit/unitFlow.cpp File Reference	73
6.30.1 Function Documentation	74
6.30.1.1 runUnitTestsFlow()	74
6.30.1.2 unitFlowDefaultConstructor()	74
6.30.1.3 unitFlowDestructor()	74
6.30.1.4 unitFlowExpression()	74
6.30.1.5 unitFlowGetName()	74
6.30.1.6 unitFlowGetSystemBegin()	74
6.30.1.7 unitFlowGetSystemEnd()	75
6.30.1.8 unitFlowGetValue()	75
6.30.1.9 unitFlowSetName()	75
6.30.1.10 unitFlowSetSystemBegin()	75
6.30.1.11 unitFlowSetSystemEnd()	75
6.30.1.12 unitFlowSetValue()	75
6.31 test/unit/unitFlow.h File Reference	76
6.31.1 Function Documentation	77
6.31.1.1 runUnitTestsFlow()	77
6.31.1.2 unitFlowDefaultConstructor()	77
6.31.1.3 unitFlowDestructor()	77
6.31.1.4 unitFlowExpression()	77
6.31.1.5 unitFlowGetName()	77
6.31.1.6 unitFlowGetSystemBegin()	78
6.31.1.7 unitFlowGetSystemEnd()	78
6.31.1.8 unitFlowGetValue()	78
6.31.1.9 unitFlowSetName()	78
6.31.1.10 unitFlowSetSystemBegin()	78
6.31.1.11 unitFlowSetSystemEnd()	78
6.31.1.12 unitFlowSetValue()	78
6.32 unitFlow.h	79
6.33 test/unit/unitModel.cpp File Reference	79

6.33.1 Function Documentation	80
6.33.1.1 runUnitTestsModel()	80
6.33.1.2 unitModelAddFlow()	80
6.33.1.3 unitModelAddSystem()	80
6.33.1.4 unitModelCreateFlow()	80
6.33.1.5 unitModelCreateModel()	80
6.33.1.6 unitModelCreateSystem()	81
6.33.1.7 unitModelDefaultConstructor()	81
6.33.1.8 unitModelDestructor()	81
6.33.1.9 unitModelGetName()	81
6.33.1.10 unitModelGetTime()	81
6.33.1.11 unitModelSetName()	81
6.33.1.12 unitModelSetTime()	81
6.33.1.13 unitModelSimulate()	81
6.34 test/unit/unitModel.h File Reference	82
6.34.1 Function Documentation	83
6.34.1.1 runUnitTestsModel()	83
6.34.1.2 unitModelAddFlow()	83
6.34.1.3 unitModelAddSystem()	83
6.34.1.4 unitModelCreateFlow()	83
6.34.1.5 unitModelCreateModel()	83
6.34.1.6 unitModelCreateSystem()	84
6.34.1.7 unitModelDefaultConstructor()	84
6.34.1.8 unitModelDestructor()	84
6.34.1.9 unitModelGetName()	84
6.34.1.10 unitModelGetTime()	84
6.34.1.11 unitModelSetName()	84
6.34.1.12 unitModelSetTime()	84
6.34.1.13 unitModelSimulate()	84
6.35 unitModel.h	85
6.36 test/unit/unitSystem.cpp File Reference	85
6.36.1 Function Documentation	86
6.36.1.1 runUnitTestsSystem()	86
6.36.1.2 unitSystemDefaultConstructor()	87
6.36.1.3 unitSystemDestructor()	87
6.36.1.4 unitSystemGetName()	87
6.36.1.5 unitSystemGetValue()	87
6.36.1.6 unitSystemSetName()	87
6.36.1.7 unitSystemSetValue()	87
6.37 test/unit/unitSystem.h File Reference	88
6.37.1 Function Documentation	89
6.37.1.1 runUnitTestsSystem()	89

6.37.1.2 unitSystemDefaultConstructor()	89
6.37.1.3 unitSystemDestructor()	89
6.37.1.4 unitSystemGetName()	89
6.37.1.5 unitSystemGetValue()	89
6.37.1.6 unitSystemSetName()	89
6.37.1.7 unitSystemSetValue()	90
6.38 unitSystem.h	90
6.39 test/unit/unitTests.cpp File Reference	90
6.39.1 Function Documentation	91
6.39.1.1 runGlobal()	91
6.40 test/unit/unitTests.h File Reference	91
6.40.1 Function Documentation	92
6.40.1.1 runGlobal()	92
6.41 unitTests.h	92
Index	93

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:

Flow	13
FlowImplementation	17
ComplexFlow	9
ExponencialFlow	11
LogisticalFlow	23
UnitTestFlow	41
UnitTestFlow2	43
Model	25
ModelImplementation	29
System	36
SystemImplementation	38

Chapter 3

Class Index

3.1 Class List

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

ComplexFlow	9
ExponencialFlow	11
Flow	13
FlowImplementation	17
LogisticalFlow	23
Model	25
ModellImplementation	29
System	36
SystemImplementation	38
UnitTestFlow	41
UnitTestFlow2	43

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	47
cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp	50
src/main.cpp	67
src/lib/Flow.h	54
src/lib/FlowImplementation.cpp	56
src/lib/FlowImplementation.h	57
src/lib/Model.h	58
src/lib/ModelImplementation.cpp	60
src/lib/ModelImplementation.h	62
src/lib/System.h	64
src/lib/SystemImplementation.cpp	65
src/lib/SystemImplementation.h	65
test/functional/FunctionalTests.cpp	70
test/functional/FunctionalTests.h	71
test/functional/main.cpp	67
test/unit/main.cpp	68
test/unit/unitFlow.cpp	73
test/unit/unitFlow.h	76
test/unit/unitModel.cpp	79
test/unit/unitModel.h	82
test/unit/unitSystem.cpp	85
test/unit/unitSystem.h	88
test/unit/unitTests.cpp	90
test/unit/unitTests.h	91

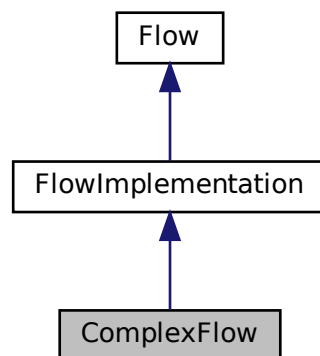
Chapter 5

Class Documentation

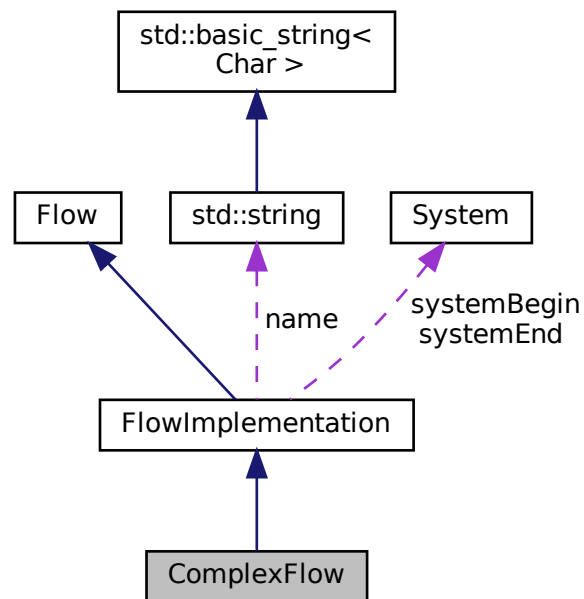
5.1 ComplexFlow Class Reference

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

Inheritance diagram for ComplexFlow:



Collaboration diagram for ComplexFlow:



Public Member Functions

- `ComplexFlow` (`std::string name`, `System *systemOut`, `System *systemIn`)
- `double expression ()` override

Additional Inherited Members

5.1.1 Detailed Description

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

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ComplexFlow()

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

Default constructor

Parameters

<i>name</i>	Initial flow name
<i>value</i>	Initial flow value
<i>systemBegin</i>	Initial system where the flow comes from
<i>systemEnd</i>	Initial system where the flow goes to

Returns

Complex flow with initial name, value, systemBegin and systemEnd

5.1.3 Member Function Documentation

5.1.3.1 expression()

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

Complex expression

Implements [FlowImplementation](#).

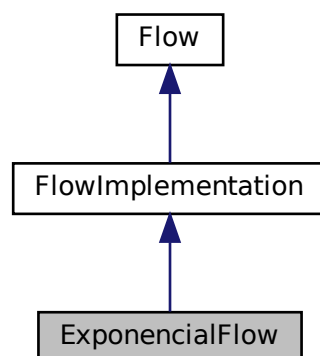
The documentation for this class was generated from the following file:

- test/functional/[FunctionalTests.h](#)

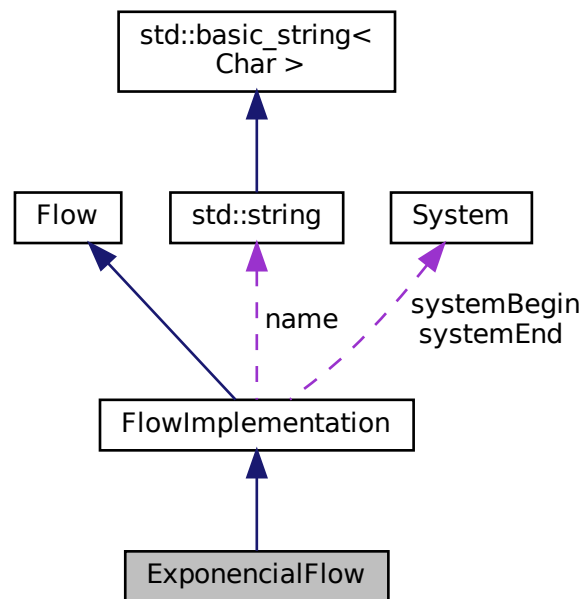
5.2 ExponentialFlow Class Reference

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

Inheritance diagram for ExponentialFlow:



Collaboration diagram for ExponentialFlow:



Public Member Functions

- [ExponentialFlow](#) (std::string *name*, [System](#) *systemOut, [System](#) *systemIn)
- double [expression](#) () override

Additional Inherited Members

5.2.1 Detailed Description

[Flow](#) that converges energy from a model to another exponentially with 1% of the initial system per timestep

5.2.2 Constructor & Destructor Documentation

5.2.2.1 ExponentialFlow()

```

ExponentialFlow::ExponentialFlow (
    std::string name,
    System * systemOut,
    System * systemIn ) [inline]
  
```

Default constructor

Parameters

<i>name</i>	Initial flow name
<i>value</i>	Initial flow value
<i>systemBegin</i>	Initial system where the flow comes from
<i>systemEnd</i>	Initial system where the flow goes to

Returns

Exponential flow with initial name, value, systemBegin and systemEnd

5.2.3 Member Function Documentation

5.2.3.1 expression()

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

Exponential expression

Implements [FlowImplementation](#).

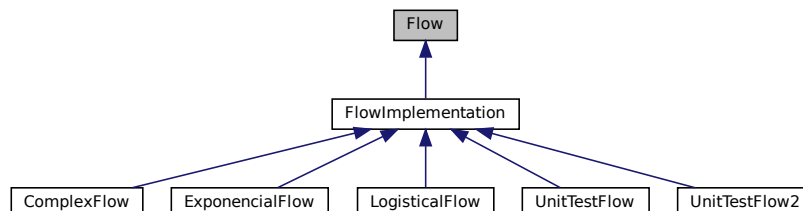
The documentation for this class was generated from the following file:

- test/functional/[FunctionalTests.h](#)

5.3 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

5.3.1 Constructor & Destructor Documentation

5.3.1.1 ~Flow()

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

Default destructor

5.3.2 Member Function Documentation

5.3.2.1 expression()

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

Sets the expression of the flow

Implemented in [ExponencialFlow](#), [LogisticalFlow](#), [ComplexFlow](#), [UnitTestFlow](#), [UnitTestFlow2](#), and [FlowImplementation](#).

5.3.2.2 getName()

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

Get system name

Implemented in [FlowImplementation](#).

5.3.2.3 `getSystemBegin()`

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

Get systemBegin

Implemented in [FlowImplementation](#).

5.3.2.4 `getSystemEnd()`

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

Get systemEnd

Implemented in [FlowImplementation](#).

5.3.2.5 `getValue()`

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

Get system value

Implemented in [FlowImplementation](#).

5.3.2.6 `setName()`

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

Set system name

Parameters

<i>n</i>	Name for the flow
----------	-------------------

Implemented in [FlowImplementation](#).

5.3.2.7 `setSystemBegin()`

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

Set systemBegin

Parameters

<i>system</i>	SystemBegin for the flow
---------------	--------------------------

Implemented in [FlowImplementation](#).

5.3.2.8 setSystemEnd()

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

Set systemBegin

Parameters

<i>system</i>	SystemEnd for the flow
---------------	------------------------

Implemented in [FlowImplementation](#).

5.3.2.9 setValue()

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

Set system value

Parameters

<i>v</i>	Value for the flow
----------	--------------------

Implemented in [FlowImplementation](#).

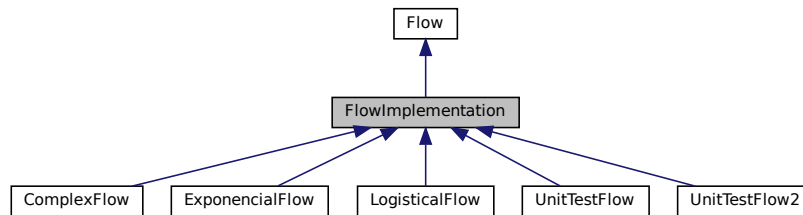
The documentation for this class was generated from the following file:

- [src/lib/Flow.h](#)

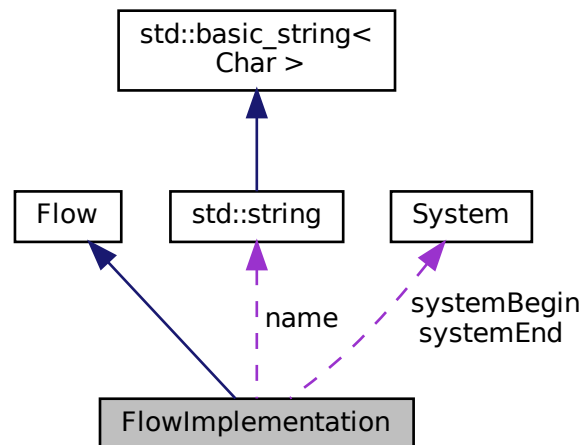
5.4 FlowImplementation Class Reference

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

Inheritance diagram for FlowImplementation:



Collaboration diagram for FlowImplementation:



Public Member Functions

- `~FlowImplementation()` override
- `FlowImplementation(std::string name, System *systemBegin, System *systemEnd)`
- `double expression()` override=0
- `std::string getName()` const override
- `void setName(std::string n)` override
- `double getValue()` const override
- `void setValue(double v)` override
- `System *getSystemBegin()` const override
- `void setSystemBegin(System *system)` override
- `System *getSystemEnd()` const override
- `void setSystemEnd(System *system)` override

Protected Attributes

- `std::string` `name`
- `double` `value`
- `System *` `systemBegin`
- `System *` `systemEnd`

5.4.1 Detailed Description

`Flow` that converges energy from a model to another

5.4.2 Constructor & Destructor Documentation

5.4.2.1 ~FlowImplementation()

```
FlowImplementation::~FlowImplementation ( ) [override], [default]
```

Default destructor

5.4.2.2 FlowImplementation()

```
FlowImplementation::FlowImplementation (
    std::string name,
    System * systemBegin,
    System * systemEnd )
```

Default constructor

Parameters

<i>name</i>	Initial flow name
<i>value</i>	Initial flow value
<i>systemBegin</i>	Initial system where the flow comes from
<i>systemEnd</i>	Initial system where the flow goes to

Returns

`Flow` with initial name, value, systemBegin and systemEnd

5.4.3 Member Function Documentation

5.4.3.1 expression()

```
double FlowImplementation::expression ( ) [override], [pure virtual]
```

Sets the expression of the flow

Implements [Flow](#).

Implemented in [ExponencialFlow](#), [LogisticalFlow](#), [ComplexFlow](#), [UnitTestFlow](#), and [UnitTestFlow2](#).

5.4.3.2 getName()

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

Get system name

Implements [Flow](#).

5.4.3.3 getSystemBegin()

```
System * FlowImplementation::getSystemBegin ( ) const [override], [virtual]
```

Get systemBegin

Implements [Flow](#).

5.4.3.4 getSystemEnd()

```
System * FlowImplementation::getSystemEnd ( ) const [override], [virtual]
```

Get systemEnd

Implements [Flow](#).

5.4.3.5 getValue()

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

Get system value

Implements [Flow](#).

5.4.3.6 setName()

```
void FlowImplementation::setName (
    std::string n ) [override], [virtual]
```

Set system name

Parameters

<i>n</i>	Name for the flow
----------	-------------------

Implements [Flow](#).

5.4.3.7 setSystemBegin()

```
void FlowImplementation::setSystemBegin (  
    System * system ) [override], [virtual]
```

Set systemBegin

Parameters

<i>system</i>	SystemBegin for the flow
---------------	--------------------------

Implements [Flow](#).

5.4.3.8 setSystemEnd()

```
void FlowImplementation::setSystemEnd (  
    System * system ) [override], [virtual]
```

Set systemBegin

Parameters

<i>system</i>	SystemEnd for the flow
---------------	------------------------

Implements [Flow](#).

5.4.3.9 setValue()

```
void FlowImplementation::setValue (  
    double v ) [override], [virtual]
```

Set system value

Parameters

<i>v</i>	Value for the flow
----------	--------------------

Implements [Flow](#).

5.4.4 Member Data Documentation

5.4.4.1 name

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

5.4.4.2 systemBegin

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

5.4.4.3 systemEnd

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

5.4.4.4 value

```
double FlowImplementation::value [protected]
```

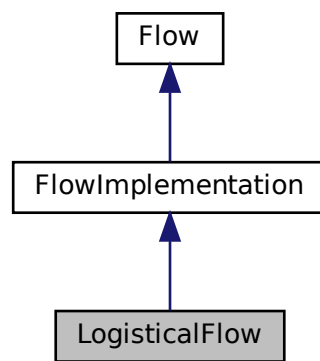
The documentation for this class was generated from the following files:

- [src/lib/FlowImplementation.h](#)
- [src/lib/FlowImplementation.cpp](#)

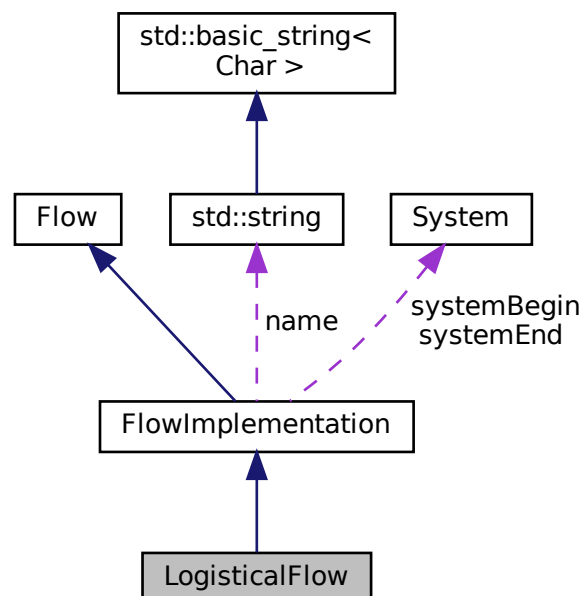
5.5 LogisticalFlow Class Reference

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

Inheritance diagram for LogisticalFlow:



Collaboration diagram for LogisticalFlow:



Public Member Functions

- [LogisticalFlow](#) (std::string *name*, [System](#) *systemOut, [System](#) *systemIn)
- double [expression](#) () override

Additional Inherited Members

5.5.1 Detailed Description

[Flow](#) that converges energy from a model to another exponentially with 1% of the end system per timestep times $\text{onde} - \text{the end system} / 70$

5.5.2 Constructor & Destructor Documentation

5.5.2.1 LogisticalFlow()

```
LogisticalFlow::LogisticalFlow (
    std::string name,
    System * systemOut,
    System * systemIn ) [inline]
```

Default constructor

Parameters

<i>name</i>	Initial flow name
<i>value</i>	Initial flow value
<i>systemBegin</i>	Initial system where the flow comes from
<i>systemEnd</i>	Initial system where the flow goes to

Returns

Logistical flow with initial name, value, systemBegin and systemEnd

5.5.3 Member Function Documentation

5.5.3.1 expression()

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

Logistical expression

Implements [FlowImplementation](#).

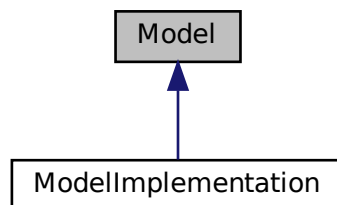
The documentation for this class was generated from the following file:

- test/functional/[FunctionalTests.h](#)

5.6 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 void `add` (System *)=0
- virtual void `add` (Flow *)=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)

5.6.1 Constructor & Destructor Documentation

5.6.1.1 ~Model()

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

Default destructor

5.6.2 Member Function Documentation

5.6.2.1 add() [1/2]

```
virtual void Model::add (
    Flow * ) [pure virtual]
```

Add a flow to the model

Parameters

<i>flow</i>	Flow to be added to the model
-------------	---

Implemented in [ModelImplementation](#).

5.6.2.2 add() [2/2]

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

Add a system to the model

Parameters

<i>system</i>	System to be added to the model
---------------	---

Implemented in [ModelImplementation](#).

5.6.2.3 createFlow()

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

5.6.2.4 createModel()

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

5.6.2.5 createSystem()

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

Implemented in [ModelImplementation](#).

5.6.2.6 endFlows()

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

Implemented in [ModelImplementation](#).

5.6.2.7 endModels()

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

Implemented in [ModelImplementation](#).

5.6.2.8 endSystems()

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

Implemented in [ModelImplementation](#).

5.6.2.9 getFlowsIterator()

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

Get model flows iterator

Implemented in [ModelImplementation](#).

5.6.2.10 getModelsIterator()

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

Get model models iterator

Implemented in [ModelImplementation](#).

5.6.2.11 getName()

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

Get model name

Implemented in [ModelImplementation](#).

5.6.2.12 getSystemsIterator()

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

Get model systems iterator

Implemented in [ModelImplementation](#).

5.6.2.13 getTime()

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

Get model time

Implemented in [ModelImplementation](#).

5.6.2.14 setName()

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

Set model name

Parameters

<i>n</i>	Name for the system
----------	---------------------

Implemented in [ModelImplementation](#).

5.6.2.15 setTime()

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

Set model time

Parameters

<i>t</i>	Name for the system
----------	---------------------

Implemented in [ModelImplementation](#).

5.6.2.16 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

<i>start</i>	Time where the simulation starts
<i>end</i>	Time where the simulation ends
<i>timestep</i>	Timestep value to simulate with

Implemented in [ModelImplementation](#).

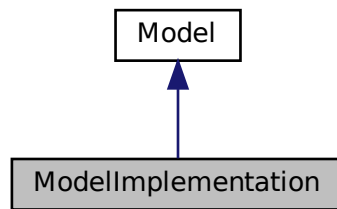
The documentation for this class was generated from the following files:

- [src/lib/Model.h](#)
- [src/lib/ModelImplementation.cpp](#)

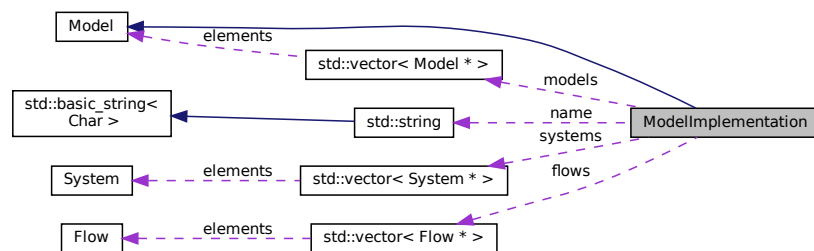
5.7 ModelImplementation Class Reference

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

Inheritance diagram for ModelImplementation:



Collaboration diagram for ModelImplementation:



Public Member Functions

- `~ModelImplementation ()` override
- `ModelImplementation (std::string name, double time)`
- `void simulate (double start, double end, double timestep)` override
- `std::string getName ()` const override
- `void setName (std::string n)` override
- `double getTime ()` const override
- `void setTime (double t)` 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
- `System * createSystem (std::string name, double value)` override

Static Public Member Functions

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

Protected Member Functions

- void [add](#) ([System](#) *system) override
- void [add](#) ([Flow](#) *flow) override

Protected Attributes

- std::string [name](#)
- double [time](#)
- std::vector< [System](#) * > [systems](#)
- std::vector< [Flow](#) * > [flows](#)

Static Protected Attributes

- static std::vector< [Model](#) * > [models](#)

5.7.1 Detailed Description

[Model](#) that simulates the energy flow through models

5.7.2 Constructor & Destructor Documentation

5.7.2.1 ~ModelImplementation()

```
ModelImplementation::~ModelImplementation ( ) [override]
```

Default destructor destrutor padrao

5.7.2.2 ModelImplementation()

```
ModelImplementation::ModelImplementation (
    std::string name,
    double time )
```

Default constructor

Parameters

<i>name</i>	Initial model name
<i>time</i>	Initial model time

Returns

[Model](#) with initial name and time

5.7.3 Member Function Documentation

5.7.3.1 `add()` [1/2]

```
void ModelImplementation::add (  
    Flow * flow ) [override], [protected], [virtual]
```

Add a flow to the model

Parameters

<i>flow</i>	Flow to be added to the model
-------------	---

Implements [Model](#).

5.7.3.2 `add()` [2/2]

```
void ModelImplementation::add (  
    System * system ) [override], [protected], [virtual]
```

Add a system to the model

Parameters

<i>system</i>	System to be added to the model
---------------	---

Implements [Model](#).

5.7.3.3 `createModel()`

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

5.7.3.4 createSystem()

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

Implements [Model](#).

5.7.3.5 endFlows()

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

Implements [Model](#).

5.7.3.6 endModels()

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

Implements [Model](#).

5.7.3.7 endSystems()

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

Implements [Model](#).

5.7.3.8 getFlowsIterator()

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

Get model flows iterator

Implements [Model](#).

5.7.3.9 getModelsIterator()

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

Get model models iterator

Implements [Model](#).

5.7.3.10 getName()

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

Get model name

Implements [Model](#).

5.7.3.11 getSystemsIterator()

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

Get model systems iterator

Implements [Model](#).

5.7.3.12 getTime()

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

Get model time

Implements [Model](#).

5.7.3.13 setName()

```
void ModelImplementation::setName (  
    std::string n ) [override], [virtual]
```

Set model name

Parameters

<i>n</i>	Name for the system
----------	---------------------

Implements [Model](#).

5.7.3.14 setTime()

```
void ModelImplementation::setTime (  
    double t ) [override], [virtual]
```

Set model time

Parameters

<i>t</i>	Name for the system
----------	---------------------

Implements [Model](#).

5.7.3.15 simulate()

```
void ModelImplementation::simulate (
    double start,
    double end,
    double timestep ) [override], [virtual]
```

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

Parameters

<i>start</i>	Time where the simulation starts
<i>end</i>	Time where the simulation ends
<i>timestep</i>	Timestep value to simulate with

Implements [Model](#).

5.7.4 Member Data Documentation

5.7.4.1 flows

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

5.7.4.2 models

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

5.7.4.3 name

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

5.7.4.4 systems

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

5.7.4.5 time

```
double ModelImplementation::time [protected]
```

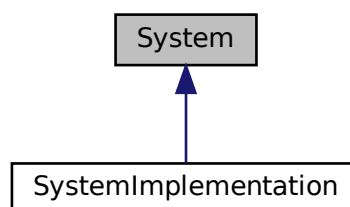
The documentation for this class was generated from the following files:

- src/lib/[ModelImplementation.h](#)
- src/lib/[ModelImplementation.cpp](#)

5.8 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

5.8.1 Constructor & Destructor Documentation

5.8.1.1 ~System()

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

Default destructor

5.8.2 Member Function Documentation

5.8.2.1 getName()

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

Get system name

Implemented in [SystemImplementation](#).

5.8.2.2 getValue()

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

Get system value

Implemented in [SystemImplementation](#).

5.8.2.3 setName()

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

Set system name

Parameters

<i>n</i>	Name for the system
----------	---------------------

Implemented in [SystemImplementation](#).

5.8.2.4 setValue()

```
virtual void System::setValue (
```

```
double ) [pure virtual]
```

Set system value

Parameters

<i>v</i>	Value for the system
----------	----------------------

Implemented in [SystemImplementation](#).

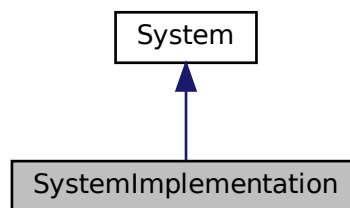
The documentation for this class was generated from the following file:

- [src/lib/System.h](#)

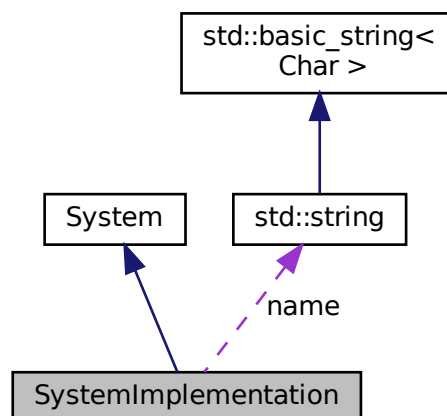
5.9 SystemImplementation Class Reference

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

Inheritance diagram for SystemImplementation:



Collaboration diagram for SystemImplementation:



Public Member Functions

- [~SystemImplementation](#) () override
- [SystemImplementation](#) (std::string [name](#), double [value](#))
- std::string [getName](#) () const override
- void [setName](#) (std::string n) override
- double [getValue](#) () const override
- void [setValue](#) (double v) override

Protected Attributes

- std::string [name](#)
- double [value](#)

5.9.1 Detailed Description

[System](#) that stores energy

5.9.2 Constructor & Destructor Documentation

5.9.2.1 ~SystemImplementation()

```
SystemImplementation::~SystemImplementation ( ) [override], [default]
```

Default destructor

5.9.2.2 SystemImplementation()

```
SystemImplementation::SystemImplementation (
    std::string name,
    double value )
```

Default constructor

Parameters

<i>name</i>	Initial system name
<i>value</i>	Initial system value

Returns

[System](#) with initial name and value

5.9.3 Member Function Documentation

5.9.3.1 getName()

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

Get system name

Implements [System](#).

5.9.3.2 getValue()

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

Get system value

Implements [System](#).

5.9.3.3 setName()

```
void SystemImplementation::setName (
    std::string n ) [override], [virtual]
```

Set system name

Parameters

<i>n</i>	Name for the system
----------	---------------------

Implements [System](#).

5.9.3.4 setValue()

```
void SystemImplementation::setValue (
    double v ) [override], [virtual]
```

Set system value

Parameters

v	Value for the system
---	----------------------

Implements [System](#).

5.9.4 Member Data Documentation

5.9.4.1 name

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

5.9.4.2 value

```
double SystemImplementation::value [protected]
```

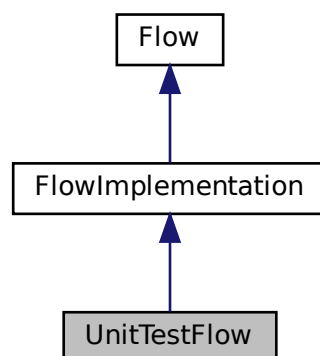
The documentation for this class was generated from the following files:

- [src/lib/SystemImplementation.h](#)
- [src/lib/SystemImplementation.cpp](#)

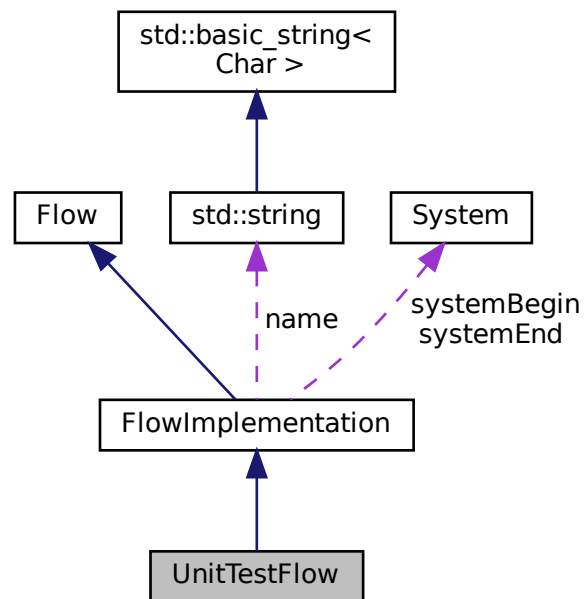
5.10 UnitTestFlow Class Reference

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

Inheritance diagram for UnitTestFlow:



Collaboration diagram for `UnitTestFixture`:



Public Member Functions

- `UnitTestFixture` (`std::string name`, `System *systemBegin`, `System *systemEnd`)
- `double expression` () override

Additional Inherited Members

5.10.1 Detailed Description

`Flow` used for testing

5.10.2 Constructor & Destructor Documentation

5.10.2.1 UnitTestFixture()

```

UnitTestFixture::UnitTestFixture (
    std::string name,
    System * systemBegin,
    System * systemEnd ) [inline]
  
```

Default constructor

Parameters

<i>name</i>	Initial flow name
<i>value</i>	Initial flow value
<i>systemBegin</i>	Initial system where the flow comes from
<i>systemEnd</i>	Initial system where the flow goes to

Returns

[UnitTestFlow](#) with initial name, value, systemBegin and systemEnd

5.10.3 Member Function Documentation

5.10.3.1 expression()

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

[Flow](#) expression method implementation for testing

Implements [FlowImplementation](#).

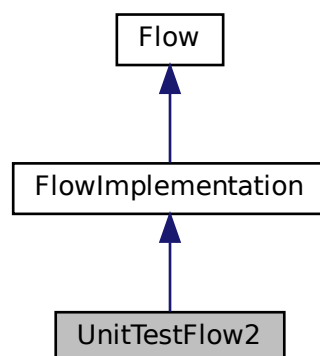
The documentation for this class was generated from the following file:

- test/unit/[unitFlow.h](#)

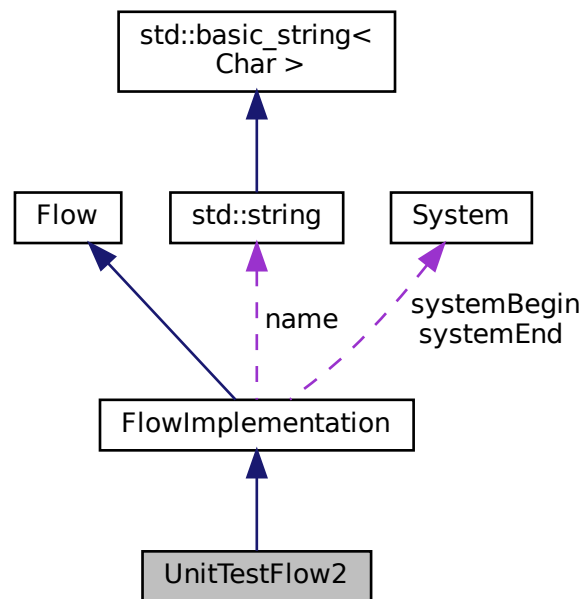
5.11 UnitTestFlow2 Class Reference

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

Inheritance diagram for UnitTestFlow2:



Collaboration diagram for `UnitTestFixture2`:



Public Member Functions

- `UnitTestFixture2` (`std::string name`, `System *systemBegin`, `System *systemEnd`)
- `double expression ()` override

Additional Inherited Members

5.11.1 Detailed Description

`Flow` used for testing

5.11.2 Constructor & Destructor Documentation

5.11.2.1 UnitTestFixture2()

```

UnitTestFixture2::UnitTestFixture2 (
    std::string name,
    System * systemBegin,
    System * systemEnd ) [inline]
  
```

Default constructor

Parameters

<i>name</i>	Initial flow name
<i>value</i>	Initial flow value
<i>systemBegin</i>	Initial system where the flow comes from
<i>systemEnd</i>	Initial system where the flow goes to

Returns

[UnitTestFlow2](#) with initial name, value, systemBegin and systemEnd

5.11.3 Member Function Documentation

5.11.3.1 `expression()`

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

[Flow](#) expression method implementation for testing

Implements [FlowImplementation](#).

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/CMakeCCompilerId.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/CMakeCXXCompilerId.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

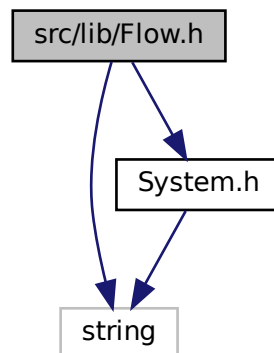
6.6 cmake-build-debug/CMakeFiles/TargetDirectories.txt File Reference

6.7 CMakeLists.txt File Reference

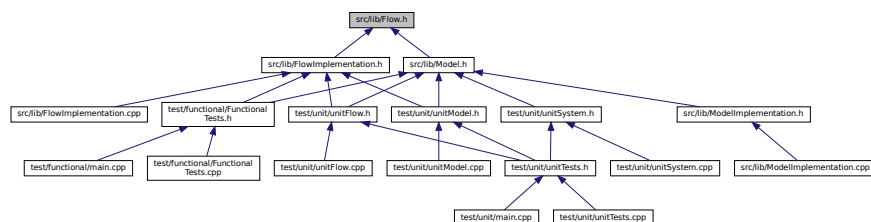
6.8 README.md File Reference

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

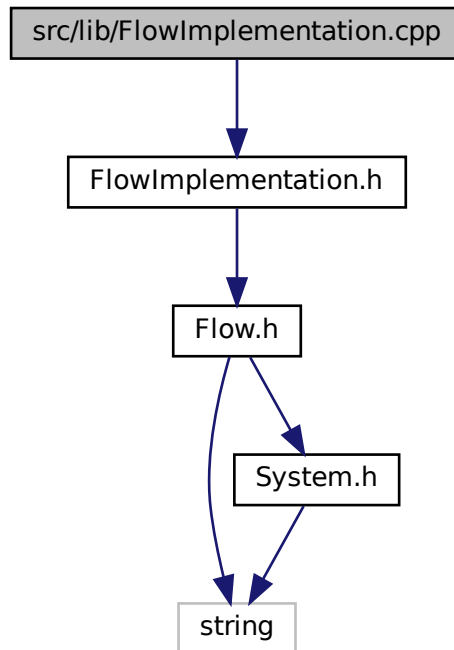
[Go to the documentation of this file.](#)

```
1 //
2 // Created by joaozenobio on 27/04/2022.
3 //
4
5 #ifndef ENGL_FLOW_H
6 #define ENGL_FLOW_H
7
8 #include <string>
9
10 #include "System.h"
11
12 class Flow {
13 public:
14     virtual ~Flow() = default;
15     virtual std::string getName() const = 0;
16     virtual void setName(std::string) = 0;
17     virtual double getValue() const = 0;
18     virtual void setValue(double) = 0;
19     virtual double expression() = 0;
20     virtual System* getSystemBegin() const = 0;
21     virtual void setSystemBegin(System*) = 0;
22     virtual System* getSystemEnd() const = 0;
23     virtual void setSystemEnd(System*) = 0;
24 };
25
26 #endif //ENGL_FLOW_H
```

6.11 src/lib/FlowImplementation.cpp File Reference

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

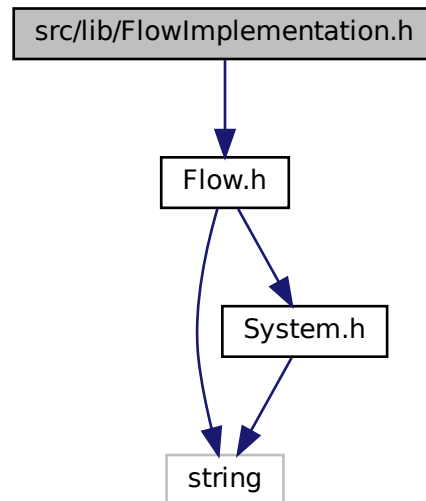
Include dependency graph for FlowImplementation.cpp:



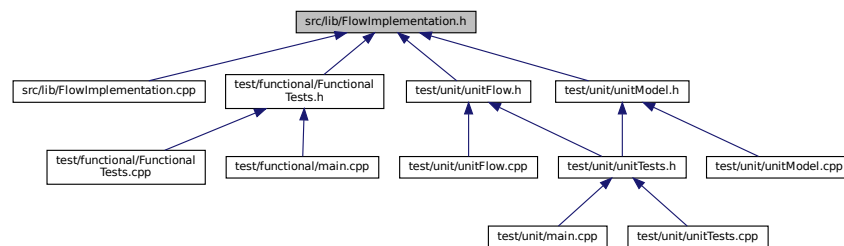
6.12 src/lib/FlowImplementation.h File Reference

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

Include dependency graph for FlowImplementation.h:



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



Classes

- class [FlowImplementation](#)

6.13 FlowImplementation.h

[Go to the documentation of this file.](#)

```

1 //
2 // Created by joaozenobio on 28/04/2022.
3 //

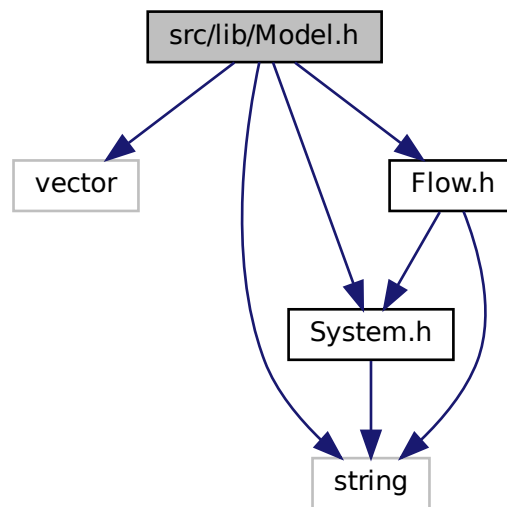
```

```
4
5 #ifndef ENGL_FLOWIMPLEMENTATION_H
6 #define ENGL_FLOWIMPLEMENTATION_H
7
8
9 #include "Flow.h"
10
11 class FlowImplementation : public Flow {
12
13 private:
14     FlowImplementation(const FlowImplementation& flow);
15
16     FlowImplementation& operator=(const FlowImplementation& flow);
17
18 protected:
19     std::string name;
20     double value;
21     System* systemBegin;
22     System* systemEnd;
23
24 public:
25     ~FlowImplementation() override;
26
27     FlowImplementation(std::string name, System* systemBegin, System* systemEnd);
28
29     double expression() override = 0;
30
31     std::string getName() const override;
32
33     void setName(std::string n) override;
34
35     double getValue() const override;
36
37     void setValue(double v) override;
38
39     System* getSystemBegin() const override;
40
41     void setSystemBegin(System* system) override;
42
43     System* getSystemEnd() const override;
44
45     void setSystemEnd(System* system) override;
46 };
47
48 #endif //ENGL_FLOWIMPLEMENTATION_H
```

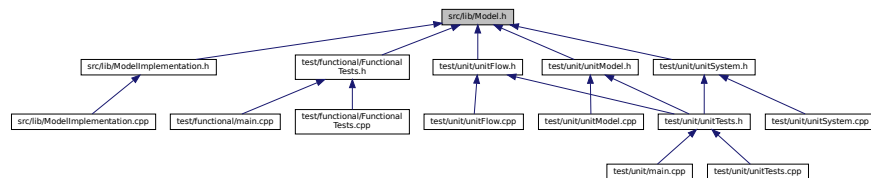
6.14 src/lib/Model.h File Reference

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

Include dependency graph for Model.h:



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



Classes

- class [Model](#)

6.15 Model.h

[Go to the documentation of this file.](#)

```

1 //
2 // Created by joaozenobio on 27/04/2022.
3 //
4
5 #ifndef ENGL_MODEL_H
6 #define ENGL_MODEL_H
7
8
9 #include <vector>
10 #include <string>
11
12 #include "System.h"
13 #include "Flow.h"

```

```

14
15 class Model {
16 public:
20     virtual ~Model() = default;
27     virtual void simulate(double, double, double) = 0;
31     virtual std::string getName() const = 0;
36     virtual void setName(std::string) = 0;
40     virtual double getTime() const = 0;
45     virtual void setTime(double) = 0;
50     virtual void add(System*) = 0;
55     virtual void add(Flow*) = 0;
59     virtual std::vector<System*>::iterator getSystemsIterator() = 0;
63     virtual std::vector<Flow*>::iterator getFlowsIterator() = 0;
67     virtual std::vector<Model*>::iterator getModelsIterator() = 0;
68
69     virtual std::vector<System*>::iterator endSystems() = 0;
70
71     virtual std::vector<Flow*>::iterator endFlows() = 0;
72
73     virtual std::vector<Model*>::iterator endModels() = 0;
74
75     virtual System* createSystem(std::string name, double value) = 0;
76
77     template<typename FlowType>
78     Flow* createFlow(std::string name, System* systemBegin, System* systemEnd){
79         Flow* flow = new FlowType(name, systemBegin, systemEnd);
80         add(flow);
81         return flow;
82     }
83
84     static Model* createModel(std::string name, double time);
85 };
86
87
88 #endif //ENGL_MODEL_H

```

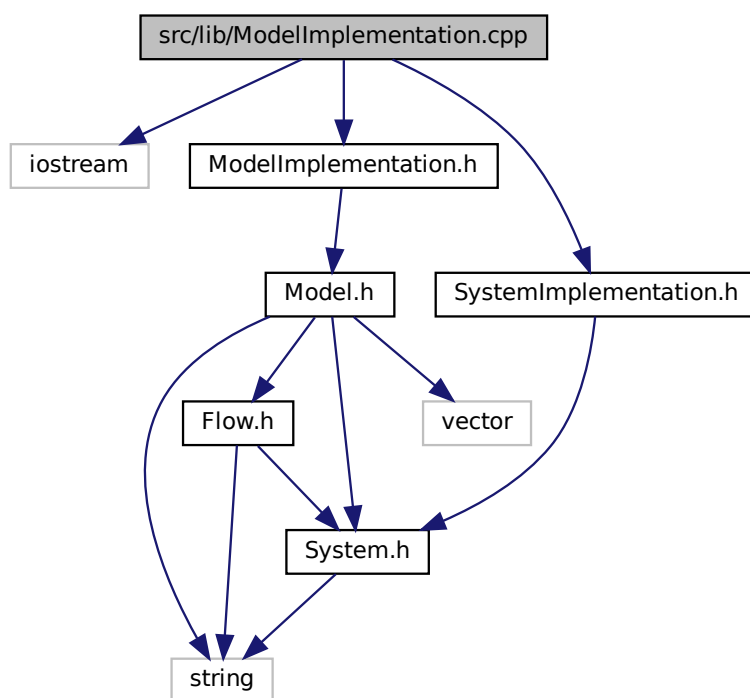
6.16 src/lib/ModelImplementation.cpp File Reference

```

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

```

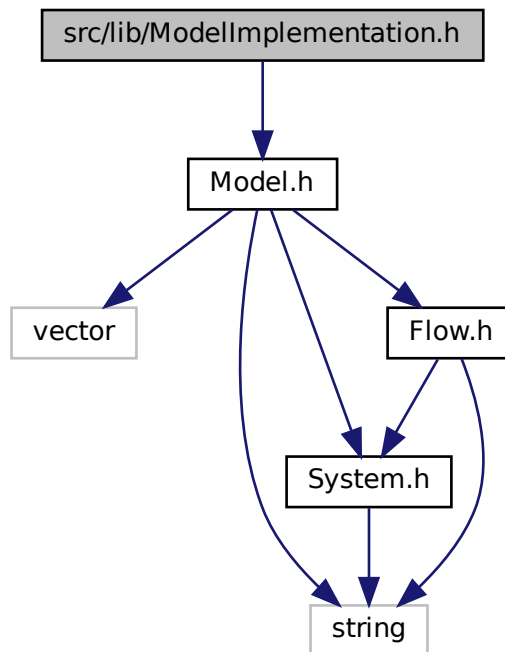
Include dependency graph for ModellImplementation.cpp:



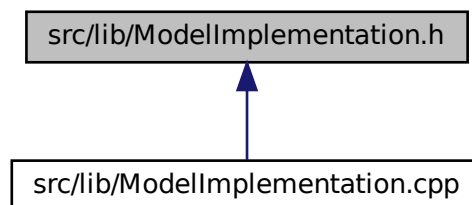
6.17 src/lib/ModellImplementation.h File Reference

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

Include dependency graph for ModellImplementation.h:



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



Classes

- class [ModellImplementation](#)

6.18 ModelImplementation.h

[Go to the documentation of this file.](#)

```

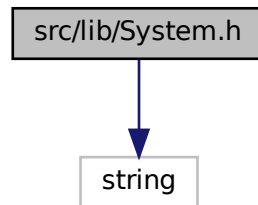
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENG1_MODELIMPLEMENTATION_H
6 #define ENG1_MODELIMPLEMENTATION_H
7
8
9 #include "Model.h"
10
11 class ModelImplementation: public Model {
12 private:
13     ModelImplementation(const ModelImplementation& model);
14
15     ModelImplementation& operator=(const ModelImplementation& model);
16
17 protected:
18     std::string name;
19     double time;
20     std::vector<System*> systems;
21     std::vector<Flow*> flows;
22     static std::vector<Model*> models;
23     void add(System* system) override;
24
25     void add(Flow* flow) override;
26
27 public:
28     ~ModelImplementation() override;
29
30     ModelImplementation(std::string name, double time);
31
32     void simulate(double start, double end, double timestep) override;
33
34     std::string getName() const override;
35
36     void setName(std::string n) override;
37
38     double getTime() const override;
39
40     void setTime(double t) override;
41
42     std::vector<System*>::iterator getSystemsIterator() override;
43
44     std::vector<Flow*>::iterator getFlowsIterator() override;
45
46     std::vector<Model*>::iterator getModelsIterator() override;
47
48     std::vector<System*>::iterator endSystems() override;
49
50     std::vector<Flow*>::iterator endFlows() override;
51
52     std::vector<Model*>::iterator endModels() override;
53
54     System* createSystem(std::string name, double value) override;
55
56     static Model* createModel(std::string name, double time);
57 };
58
59 #endif //ENG1_MODELIMPLEMENTATION_H

```

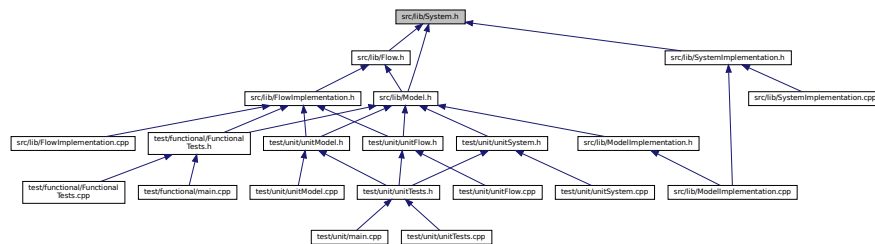
6.19 src/lib/System.h File Reference

```
#include <string>
```

Include dependency graph for System.h:



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



Classes

- class [System](#)

6.20 System.h

[Go to the documentation of this file.](#)

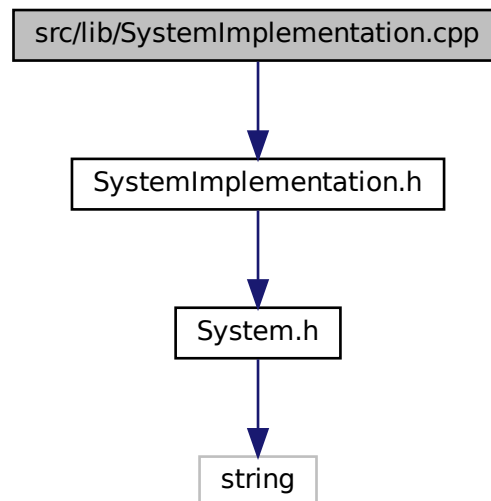
```

1 //
2 // Created by joaozenobio on 27/04/2022.
3 //
4
5 #ifndef ENGL_SYSTEM_H
6 #define ENGL_SYSTEM_H
7
8 #include <string>
9
10 class System {
11 public:
12     virtual ~System() = default;
13     virtual std::string getName() const = 0;
14     virtual void setName(std::string) = 0;
15     virtual double getValue() const = 0;
16     virtual void setValue(double) = 0;
17 };
18
19 #endif //ENGL_SYSTEM_H
  
```

6.21 src/lib/SystemImplementation.cpp File Reference

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

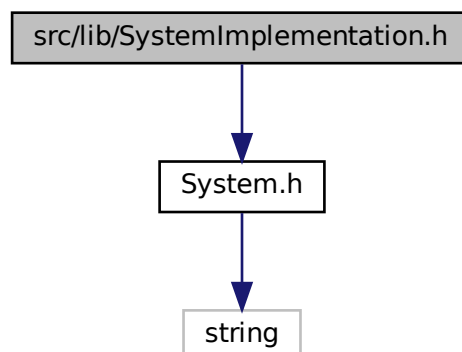
Include dependency graph for SystemImplementation.cpp:



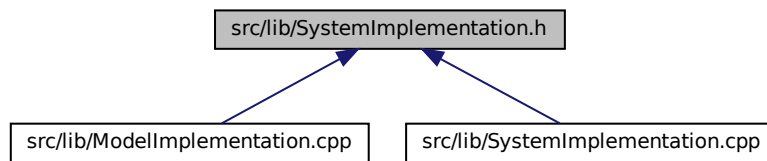
6.22 src/lib/SystemImplementation.h File Reference

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

Include dependency graph for SystemImplementation.h:



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



Classes

- class [SystemImplementation](#)

6.23 SystemImplementation.h

[Go to the documentation of this file.](#)

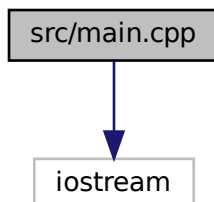
```

1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENGL_SYSTEMIMPLEMENTATION_H
6 #define ENGL_SYSTEMIMPLEMENTATION_H
7
8 #include "System.h"
9
10
11
12
13 class SystemImplementation : public System {
14
15 private:
16     SystemImplementation(const SystemImplementation& system);
17
18     SystemImplementation& operator=(const SystemImplementation& system);
19
20 protected:
21     std::string name;
22     double value;
23
24 public:
25     ~SystemImplementation() override;
26
27     SystemImplementation(std::string name, double value);
28
29     std::string getName() const override;
30
31     void setName(std::string n) override;
32
33     double getValue() const override;
34
35     void setValue(double v) override;
36 };
37
38 #endif //ENGL_SYSTEMIMPLEMENTATION_H
  
```

6.24 src/main.cpp File Reference

```
#include <iostream>
```

Include dependency graph for main.cpp:



Functions

- int `main` ()

6.24.1 Function Documentation

6.24.1.1 main()

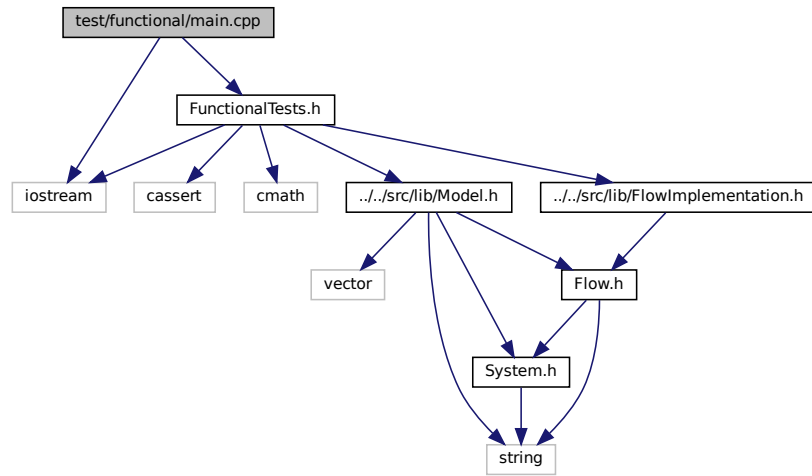
```
int main ( )
```

6.25 test/functional/main.cpp File Reference

```
#include <iostream>
```

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

Include dependency graph for main.cpp:



Functions

- `int main()`

6.25.1 Function Documentation

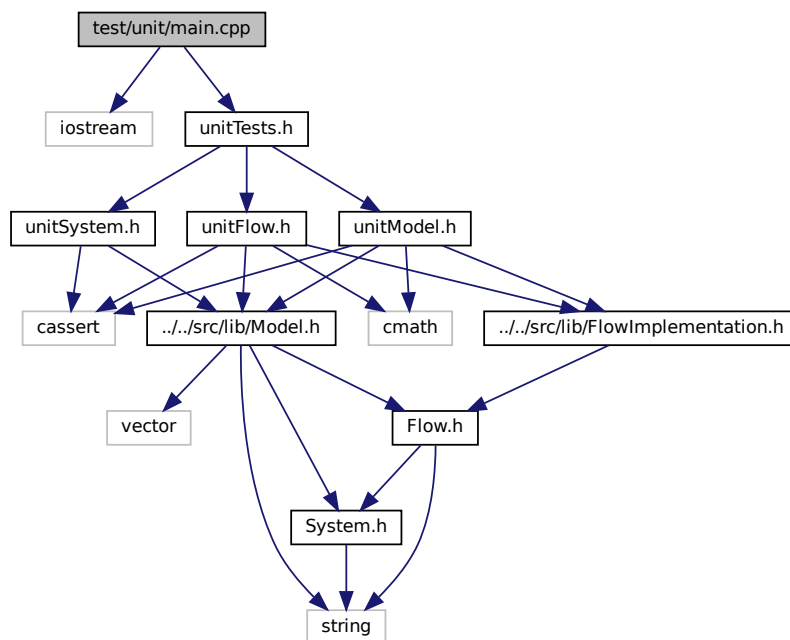
6.25.1.1 main()

```
int main ( )
```

6.26 test/unit/main.cpp File Reference

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

Include dependency graph for main.cpp:



Functions

- `int main ()`

6.26.1 Function Documentation

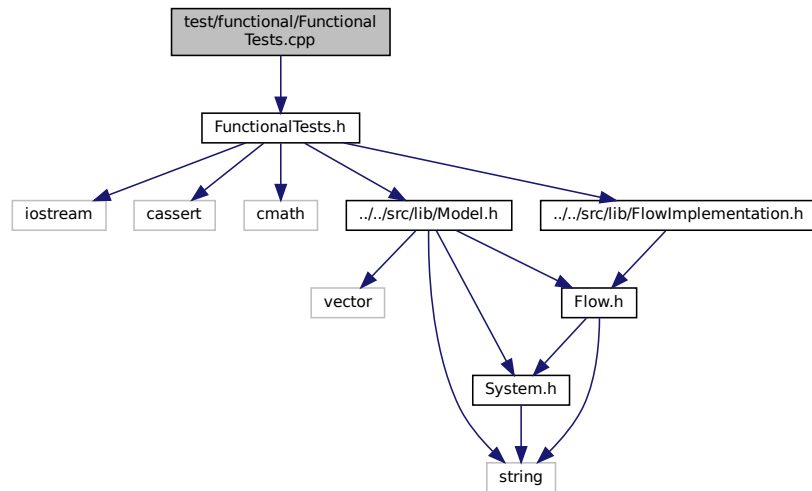
6.26.1.1 `main()`

```
int main ( )
```

6.27 test/functional/FunctionalTests.cpp File Reference

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

Include dependency graph for FunctionalTests.cpp:



Functions

- void [ExponencialTest](#) ()
- void [LogisticalTest](#) ()
- void [ComplexTest](#) ()

6.27.1 Function Documentation

6.27.1.1 ComplexTest()

```
void ComplexTest ( )
```

Function to test the complex flow.

6.27.1.2 ExponencialTest()

```
void ExponencialTest ( )
```

Function to test the exponencial flow.

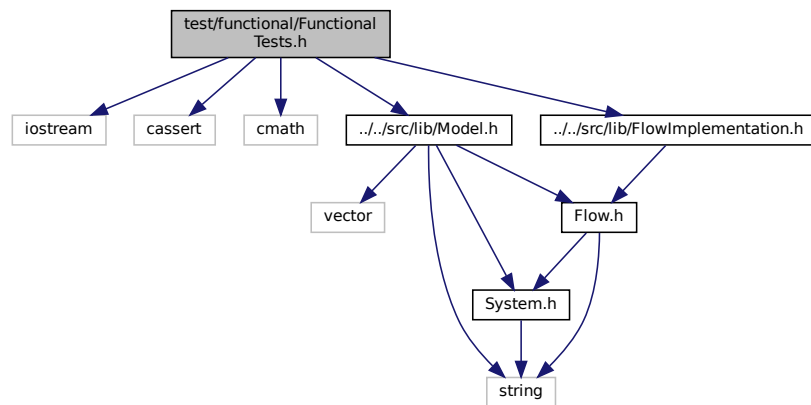
6.27.1.3 LogisticalTest()

```
void LogisticalTest ( )
```

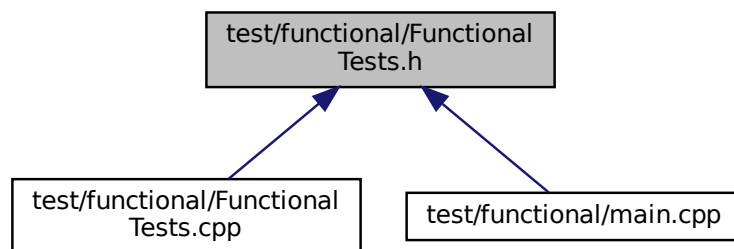
Function to test the logistical flow.

6.28 test/functional/FunctionalTests.h File Reference

```
#include <iostream>
#include <cassert>
#include <cmath>
#include "../src/lib/Model.h"
#include "../src/lib/FlowImplementation.h"
Include dependency graph for FunctionalTests.h:
```



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



Classes

- class [ExponencialFlow](#)
- class [LogisticalFlow](#)
- class [ComplexFlow](#)

Functions

- void [ExponencialTest](#) ()
- void [ComplexTest](#) ()
- void [LogisticalTest](#) ()

6.28.1 Function Documentation

6.28.1.1 ComplexTest()

```
void ComplexTest ( )
```

Function to test the complex flow.

6.28.1.2 ExponencialTest()

```
void ExponencialTest ( )
```

Function to test the exponencial flow.

6.28.1.3 LogisticalTest()

```
void LogisticalTest ( )
```

Function to test the logistical flow.

6.29 FunctionalTests.h

[Go to the documentation of this file.](#)

```
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENGL_FUNCTIONALTESTS_H
6 #define ENGL_FUNCTIONALTESTS_H
7
8 #include <iostream>
9 #include <cassert>
10 #include <cmath>
11
12 #include "../src/lib/Model.h"
13 #include "../src/lib/FlowImplementation.h"
14
15 class ExponencialFlow : public FlowImplementation{
16 public:
17     ExponencialFlow(std::string name, System* systemOut, System* systemIn) : FlowImplementation(name,
18         systemOut, systemIn) {}
19     double expression() override {
20         return 0.01 * getSystemBegin()->getValue();
21     }
22 };
23
24 class LogisticalFlow : public FlowImplementation{
25 public:
```

```

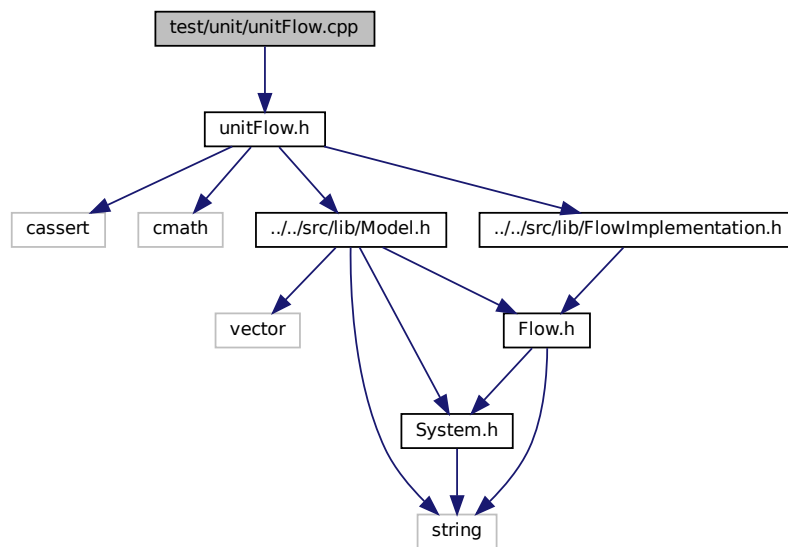
51     LogisticalFlow(std::string name, System* systemOut, System* systemIn) : FlowImplementation(name,
    systemOut, systemIn) {}
55     double expression() override {
56         return 0.01 * this->getSystemEnd()->getValue() * (1 - this->getSystemEnd()->getValue() / 70);
57     }
58 };
59
63 class ComplexFlow : public FlowImplementation{
64 public:
73     ComplexFlow(std::string name, System* systemOut, System* systemIn) : FlowImplementation(name,
    systemOut, systemIn) {}
77     double expression() override {
78         return 0.01 * getSystemBegin()->getValue();
79     }
80 };
81
82 void ExponentialTest();
83 void ComplexTest();
84 void LogisticalTest();
85
86 #endif //ENGL_FUNCTIONALTESTS_H

```

6.30 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](#) ()

6.30.1 Function Documentation

6.30.1.1 [runUnitTestsFlow](#)()

```
void runUnitTestsFlow ( )
```

Run all unit tests for [Flow](#)

6.30.1.2 [unitFlowDefaultConstructor](#)()

```
void unitFlowDefaultConstructor ( )
```

Tests [Flow](#) default constructor

6.30.1.3 [unitFlowDestructor](#)()

```
void unitFlowDestructor ( )
```

Tests [Flow](#) destructor

6.30.1.4 [unitFlowExpression](#)()

```
void unitFlowExpression ( )
```

Tests [Flow](#) expression

6.30.1.5 [unitFlowGetName](#)()

```
void unitFlowGetName ( )
```

Tests [Flow](#) getName method

6.30.1.6 [unitFlowGetSystemBegin](#)()

```
void unitFlowGetSystemBegin ( )
```

Tests [Flow](#) getSystemBegin method

6.30.1.7 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests [Flow](#) getSystemEnd method

6.30.1.8 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests [Flow](#) getValue method

6.30.1.9 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests [Flow](#) setName method

6.30.1.10 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests [Flow](#) setSystemBegin method

6.30.1.11 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests [Flow](#) setSystemEnd method

6.30.1.12 unitFlowSetValue()

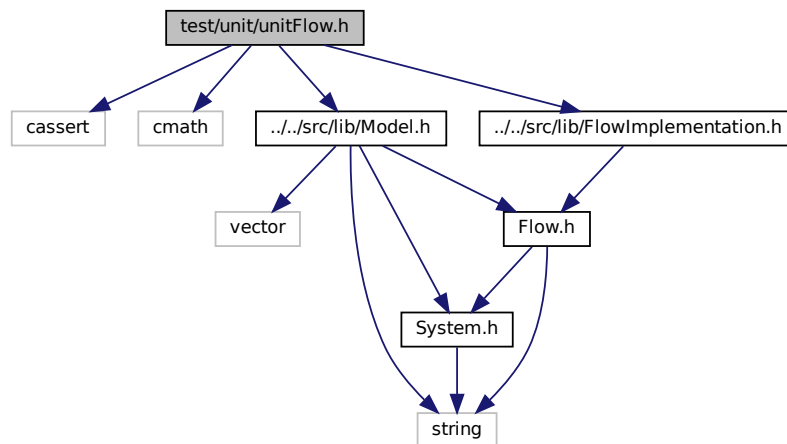
```
void unitFlowSetValue ( )
```

Tests [Flow](#) setValue method

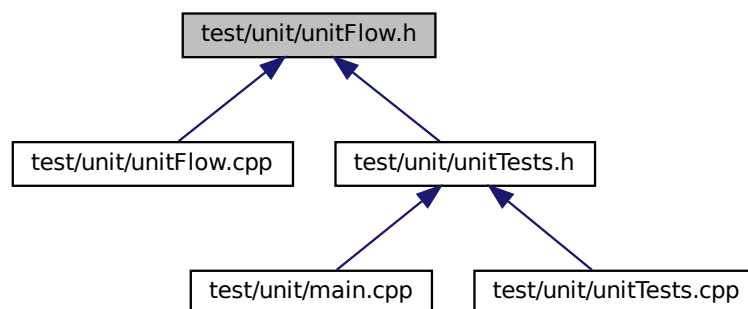
6.31 test/unit/unitFlow.h File Reference

```
#include <cassert>
#include <cmath>
#include "../src/lib/Model.h"
#include "../src/lib/FlowImplementation.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](#) ()

6.31.1 Function Documentation

6.31.1.1 [runUnitTestsFlow\(\)](#)

```
void runUnitTestsFlow ( )
```

Run all unit tests for [Flow](#)

6.31.1.2 [unitFlowDefaultConstructor\(\)](#)

```
void unitFlowDefaultConstructor ( )
```

Tests [Flow](#) default constructor

6.31.1.3 [unitFlowDestructor\(\)](#)

```
void unitFlowDestructor ( )
```

Tests [Flow](#) destructor

6.31.1.4 [unitFlowExpression\(\)](#)

```
void unitFlowExpression ( )
```

Tests [Flow](#) expression

6.31.1.5 [unitFlowGetName\(\)](#)

```
void unitFlowGetName ( )
```

Tests [Flow](#) getName method

6.31.1.6 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests [Flow](#) getSystemBegin method

6.31.1.7 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests [Flow](#) getSystemEnd method

6.31.1.8 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests [Flow](#) getValue method

6.31.1.9 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests [Flow](#) setName method

6.31.1.10 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests [Flow](#) setSystemBegin method

6.31.1.11 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests [Flow](#) setSystemEnd method

6.31.1.12 unitFlowSetValue()

```
void unitFlowSetValue ( )
```

Tests [Flow](#) setValue method

6.32 unitFlow.h

[Go to the documentation of this file.](#)

```

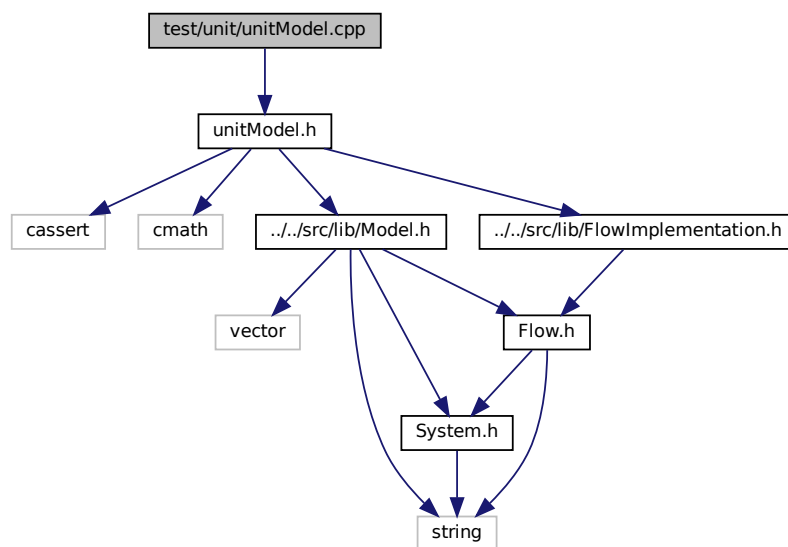
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENGL_UNITFLOW_H
6 #define ENGL_UNITFLOW_H
7
8
9 #include <cassert>
10 #include <cmath>
11
12 #include "../src/lib/Model.h"
13 #include "../src/lib/FlowImplementation.h"
14
15 class UnitTestFlow : public FlowImplementation{
16 public:
17     UnitTestFlow(std::string name, System* systemBegin, System* systemEnd) : FlowImplementation(name,
18         systemBegin, systemEnd) {}
19
20     double expression() override {
21         return 0.01 * getSystemBegin()->getValue();
22     }
23 };
24
25 void unitFlowDestructor();
26 void unitFlowDefaultConstructor();
27 void unitFlowExpression();
28 void unitFlowGetName();
29 void unitFlowSetName();
30 void unitFlowGetValue();
31 void unitFlowSetValue();
32 void unitFlowGetSystemBegin();
33 void unitFlowSetSystemBegin();
34 void unitFlowGetSystemEnd();
35 void unitFlowSetSystemEnd();
36 void runUnitTestsFlow();
37
38 #endif //ENGL_UNITFLOW_H

```

6.33 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](#) ()

6.33.1 Function Documentation

6.33.1.1 [runUnitTestsModel\(\)](#)

```
void runUnitTestsModel ( )
```

Run all unit tests for [Model](#)

6.33.1.2 [unitModelAddFlow\(\)](#)

```
void unitModelAddFlow ( )
```

Tests [Model](#) add to add a [Flow](#)

6.33.1.3 [unitModelAddSystem\(\)](#)

```
void unitModelAddSystem ( )
```

Tests [Model](#) add to add a [System](#)

6.33.1.4 [unitModelCreateFlow\(\)](#)

```
void unitModelCreateFlow ( )
```

Tests [Model](#) createFlow

6.33.1.5 [unitModelCreateModel\(\)](#)

```
void unitModelCreateModel ( )
```

Tests [Model](#) createModel

6.33.1.6 unitModelCreateSystem()

```
void unitModelCreateSystem ( )
```

Tests [Model](#) createSystem

6.33.1.7 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests [Model](#) default constructor

6.33.1.8 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests [Model](#) destructor

6.33.1.9 unitModelGetName()

```
void unitModelGetName ( )
```

Tests [Model](#) getName method

6.33.1.10 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests [Model](#) getTime method

6.33.1.11 unitModelSetName()

```
void unitModelSetName ( )
```

Tests [Model](#) setName method

6.33.1.12 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests [Model](#) setTime method

6.33.1.13 unitModelSimulate()

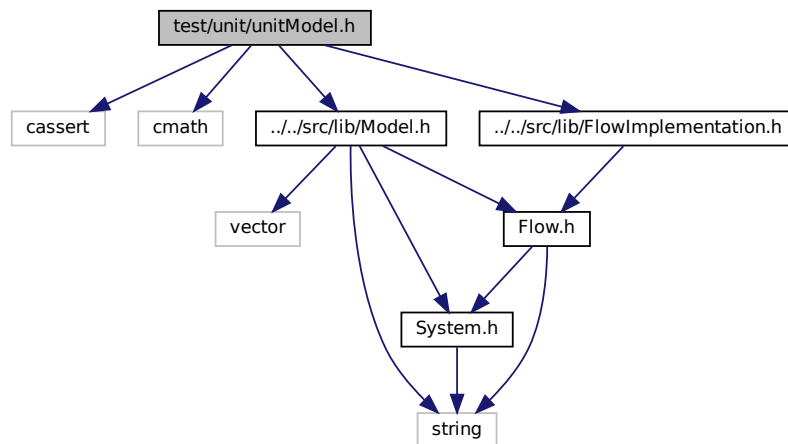
```
void unitModelSimulate ( )
```

Tests [Model](#) simulate method

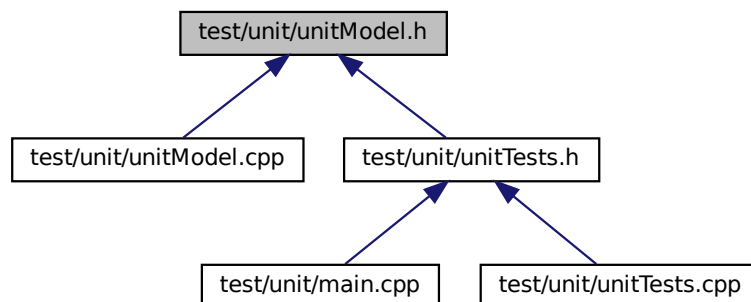
6.34 test/unit/unitModel.h File Reference

```
#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](#) ()

6.34.1 Function Documentation

6.34.1.1 [runUnitTestsModel\(\)](#)

```
void runUnitTestsModel ( )
```

Run all unit tests for [Model](#)

6.34.1.2 [unitModelAddFlow\(\)](#)

```
void unitModelAddFlow ( )
```

Tests [Model](#) add to add a [Flow](#)

6.34.1.3 [unitModelAddSystem\(\)](#)

```
void unitModelAddSystem ( )
```

Tests [Model](#) add to add a [System](#)

6.34.1.4 [unitModelCreateFlow\(\)](#)

```
void unitModelCreateFlow ( )
```

Tests [Model](#) createFlow

6.34.1.5 [unitModelCreateModel\(\)](#)

```
void unitModelCreateModel ( )
```

Tests [Model](#) createModel

6.34.1.6 unitModelCreateSystem()

```
void unitModelCreateSystem ( )
```

Tests [Model](#) createSystem

6.34.1.7 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests [Model](#) default constructor

6.34.1.8 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests [Model](#) destructor

6.34.1.9 unitModelGetName()

```
void unitModelGetName ( )
```

Tests [Model](#) getName method

6.34.1.10 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests [Model](#) getTime method

6.34.1.11 unitModelSetName()

```
void unitModelSetName ( )
```

Tests [Model](#) setName method

6.34.1.12 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests [Model](#) setTime method

6.34.1.13 unitModelSimulate()

```
void unitModelSimulate ( )
```

Tests [Model](#) simulate method

6.35 unitModel.h

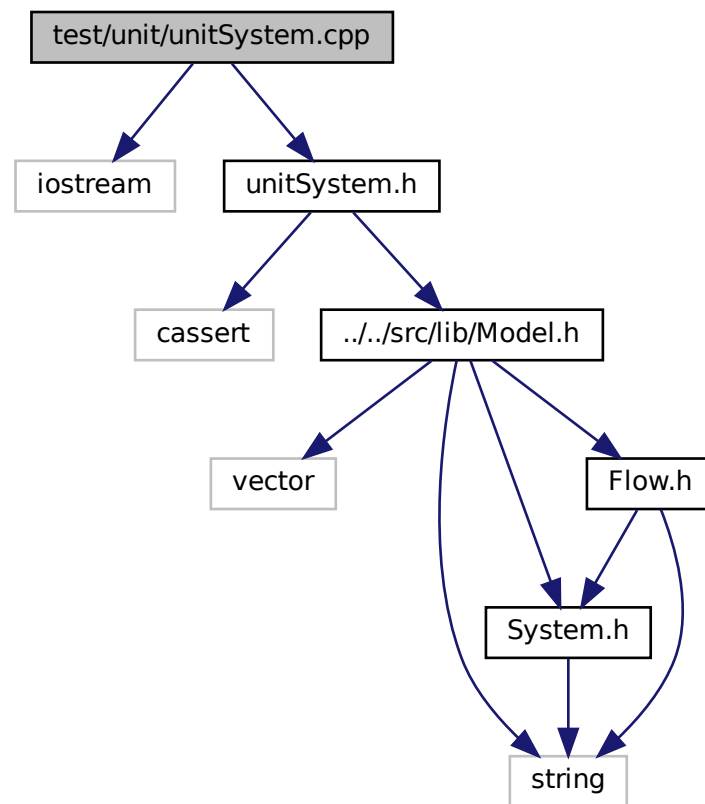
[Go to the documentation of this file.](#)

```
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENGL_UNITMODEL_H
6 #define ENGL_UNITMODEL_H
7
8
9 #include <cassert>
10 #include <cmath>
11
12 #include "../src/lib/Model.h"
13 #include "../src/lib/FlowImplementation.h"
14
15 class UnitTestFlow2 : public FlowImplementation{
16 public:
17     UnitTestFlow2(std::string name, System* systemBegin, System* systemEnd) : FlowImplementation(name,
18         systemBegin, systemEnd) {}
19     double expression() override {
20         return 0.01 * getSystemBegin()->getValue();
21     }
22 };
23
24 void unitModelDestructor();
25 void unitModelDefaultConstructor();
26 void unitModelSimulate();
27 void unitModelGetName();
28 void unitModelSetName();
29 void unitModelGetTime();
30 void unitModelSetTime();
31 void unitModelAddSystem();
32 void unitModelAddFlow();
33 void unitModelCreateSystem();
34 void unitModelCreateFlow();
35 void unitModelCreateModel();
36 void runUnitTestsModel();
37
38 #endif //ENGL_UNITMODEL_H
```

6.36 test/unit/unitSystem.cpp File Reference

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

Include dependency graph for unitSystem.cpp:



Functions

- void [unitSystemDestructor](#) ()
- void [unitSystemDefaultConstructor](#) ()
- void [unitSystemGetName](#) ()
- void [unitSystemSetName](#) ()
- void [unitSystemGetValue](#) ()
- void [unitSystemSetValue](#) ()
- void [runUnitTestsSystem](#) ()

6.36.1 Function Documentation

6.36.1.1 runUnitTestsSystem()

```
void runUnitTestsSystem ( )
```

Run all unit tests for [System](#)

6.36.1.2 unitSystemDefaultConstructor()

```
void unitSystemDefaultConstructor ( )
```

Tests [System](#) default constructor

6.36.1.3 unitSystemDestructor()

```
void unitSystemDestructor ( )
```

Tests [System](#) destructor

6.36.1.4 unitSystemGetName()

```
void unitSystemGetName ( )
```

Tests [System](#) getName method

6.36.1.5 unitSystemGetValue()

```
void unitSystemGetValue ( )
```

Tests [System](#) getValue method

6.36.1.6 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests [System](#) setName method

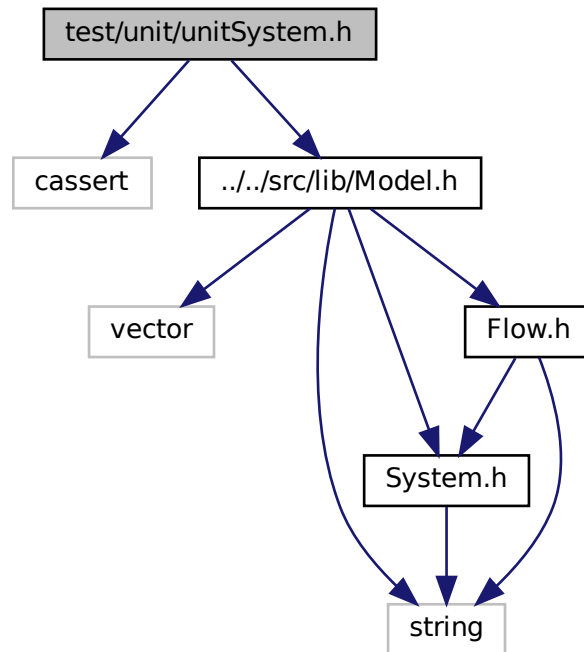
6.36.1.7 unitSystemSetValue()

```
void unitSystemSetValue ( )
```

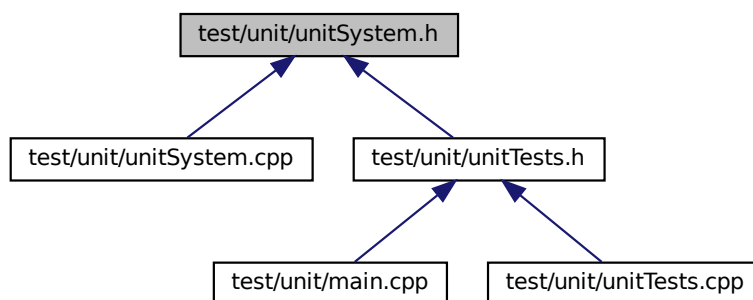
Tests [System](#) setValue method

6.37 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](#) ()

6.37.1 Function Documentation

6.37.1.1 [runUnitTestsSystem\(\)](#)

```
void runUnitTestsSystem ( )
```

Run all unit tests for [System](#)

6.37.1.2 [unitSystemDefaultConstructor\(\)](#)

```
void unitSystemDefaultConstructor ( )
```

Tests [System](#) default constructor

6.37.1.3 [unitSystemDestructor\(\)](#)

```
void unitSystemDestructor ( )
```

Tests [System](#) destructor

6.37.1.4 [unitSystemGetName\(\)](#)

```
void unitSystemGetName ( )
```

Tests [System](#) getName method

6.37.1.5 [unitSystemGetValue\(\)](#)

```
void unitSystemGetValue ( )
```

Tests [System](#) getValue method

6.37.1.6 [unitSystemSetName\(\)](#)

```
void unitSystemSetName ( )
```

Tests [System](#) setName method

6.37.1.7 unitSystemSetValue()

```
void unitSystemSetValue ( )
```

Tests [System](#) setValue method

6.38 unitSystem.h

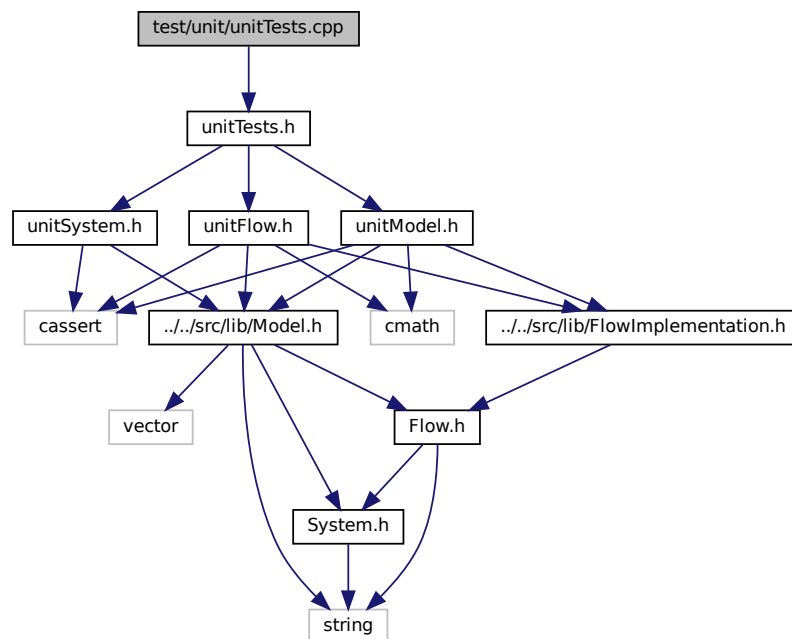
[Go to the documentation of this file.](#)

```
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENGL_UNITSYSTEM_H
6 #define ENGL_UNITSYSTEM_H
7
8 #include <cassert>
9
10 #include "../../src/lib/Model.h"
11
12 void unitSystemDestructor();
13 void unitSystemDefaultConstructor();
14 void unitSystemGetName();
15 void unitSystemSetName();
16 void unitSystemGetValue();
17 void unitSystemSetValue();
18 void runUnitTestsSystem();
19
20 #endif //ENGL_UNITSYSTEM_H
```

6.39 test/unit/unitTests.cpp File Reference

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

Include dependency graph for unitTests.cpp:



Functions

- void [runGlobal](#) ()

6.39.1 Function Documentation

6.39.1.1 runGlobal()

```
void runGlobal ( )
```

Tests [System](#) methods

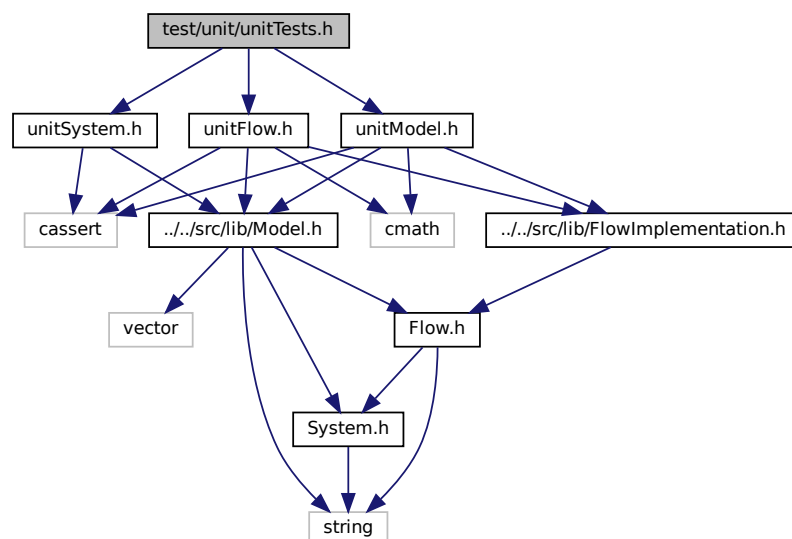
Tests [Flow](#) methods

Tests [Model](#) methods

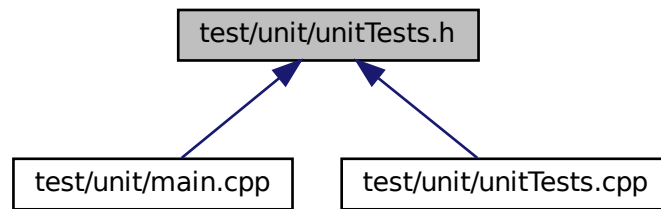
6.40 test/unit/unitTests.h File Reference

```
#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](#) ()

6.40.1 Function Documentation

6.40.1.1 runGlobal()

```
void runGlobal ( )
```

Tests [System](#) methods

Tests [Flow](#) methods

Tests [Model](#) methods

6.41 unitTests.h

[Go to the documentation of this file.](#)

```
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENGL_UNITTESTS_H
6 #define ENGL_UNITTESTS_H
7
8
9 #include "unitModel.h"
10 #include "unitFlow.h"
11 #include "unitSystem.h"
12
13 using namespace std;
14
15 void runGlobal();
16
17
18 #endif //ENGL_UNITTESTS_H
```

Index

- `__has_include`
 - `CMakeCCompilerId.c`, [47](#)
 - `CMakeCXXCompilerId.cpp`, [51](#)
 - `~Flow`
 - `Flow`, [14](#)
 - `~FlowImplementation`
 - `FlowImplementation`, [19](#)
 - `~Model`
 - `Model`, [25](#)
 - `~ModelImplementation`
 - `ModelImplementation`, [31](#)
 - `~System`
 - `System`, [36](#)
 - `~SystemImplementation`
 - `SystemImplementation`, [39](#)
- `add`
 - `Model`, [26](#)
 - `ModelImplementation`, [32](#)
- `ARCHITECTURE_ID`
 - `CMakeCCompilerId.c`, [48](#)
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `C_VERSION`
 - `CMakeCCompilerId.c`, [48](#)
- `cmake-build-debug/CMakeCache.txt`, [47](#)
- `cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c`, [47](#)
- `cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp`, [50](#)
- `cmake-build-debug/CMakeFiles/clion-environment.txt`, [54](#)
- `cmake-build-debug/CMakeFiles/clion-log.txt`, [54](#)
- `cmake-build-debug/CMakeFiles/TargetDirectories.txt`, [54](#)
- `CMakeCCompilerId.c`
 - `__has_include`, [47](#)
 - `ARCHITECTURE_ID`, [48](#)
 - `C_VERSION`, [48](#)
 - `COMPILER_ID`, [48](#)
 - `DEC`, [48](#)
 - `HEX`, [48](#)
 - `info_arch`, [49](#)
 - `info_compiler`, [49](#)
 - `info_language_extensions_default`, [49](#)
 - `info_language_standard_default`, [50](#)
 - `info_platform`, [50](#)
 - `main`, [49](#)
 - `PLATFORM_ID`, [48](#)
 - `STRINGIFY`, [49](#)
- `CMakeCXXCompilerId.cpp`
 - `__has_include`, [51](#)
 - `ARCHITECTURE_ID`, [51](#)
 - `COMPILER_ID`, [51](#)
 - `CXX_STD`, [51](#)
 - `DEC`, [51](#)
 - `HEX`, [51](#)
 - `info_arch`, [52](#)
 - `info_compiler`, [53](#)
 - `info_language_extensions_default`, [53](#)
 - `info_language_standard_default`, [53](#)
 - `info_platform`, [53](#)
 - `main`, [52](#)
 - `PLATFORM_ID`, [52](#)
 - `STRINGIFY`, [52](#)
 - `STRINGIFY_HELPER`, [52](#)
- `CMakeLists.txt`, [54](#)
- `COMPILER_ID`
 - `CMakeCCompilerId.c`, [48](#)
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `ComplexFlow`, [9](#)
 - `ComplexFlow`, [10](#)
 - `expression`, [11](#)
- `ComplexTest`
 - `FunctionalTests.cpp`, [70](#)
 - `FunctionalTests.h`, [72](#)
- `createFlow`
 - `CMakeCXXCompilerId.cpp`, [50](#)
 - `Model`, [26](#)
- `createModel`
 - `Model`, [26](#)
 - `ModelImplementation`, [32](#)
- `createSystem`
 - `Model`, [27](#)
 - `ModelImplementation`, [32](#)
- `CXX_STD`
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `DEC`
 - `CMakeCCompilerId.c`, [48](#)
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `endFlows`
 - `Model`, [27](#)
 - `ModelImplementation`, [33](#)
- `endModels`
 - `Model`, [27](#)
 - `ModelImplementation`, [33](#)
- `endSystems`
 - `Model`, [27](#)
- `STRINGIFY_HELPER`, [49](#)
- `CMakeCXXCompilerId.cpp`
 - `__has_include`, [51](#)
 - `ARCHITECTURE_ID`, [51](#)
 - `COMPILER_ID`, [51](#)
 - `CXX_STD`, [51](#)
 - `DEC`, [51](#)
 - `HEX`, [51](#)
 - `info_arch`, [52](#)
 - `info_compiler`, [53](#)
 - `info_language_extensions_default`, [53](#)
 - `info_language_standard_default`, [53](#)
 - `info_platform`, [53](#)
 - `main`, [52](#)
 - `PLATFORM_ID`, [52](#)
 - `STRINGIFY`, [52](#)
 - `STRINGIFY_HELPER`, [52](#)
- `CMakeLists.txt`, [54](#)
- `COMPILER_ID`
 - `CMakeCCompilerId.c`, [48](#)
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `ComplexFlow`, [9](#)
 - `ComplexFlow`, [10](#)
 - `expression`, [11](#)
- `ComplexTest`
 - `FunctionalTests.cpp`, [70](#)
 - `FunctionalTests.h`, [72](#)
- `createFlow`
 - `CMakeCXXCompilerId.cpp`, [50](#)
 - `Model`, [26](#)
- `createModel`
 - `Model`, [26](#)
 - `ModelImplementation`, [32](#)
- `createSystem`
 - `Model`, [27](#)
 - `ModelImplementation`, [32](#)
- `CXX_STD`
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `DEC`
 - `CMakeCCompilerId.c`, [48](#)
 - `CMakeCXXCompilerId.cpp`, [51](#)
- `endFlows`
 - `Model`, [27](#)
 - `ModelImplementation`, [33](#)
- `endModels`
 - `Model`, [27](#)
 - `ModelImplementation`, [33](#)
- `endSystems`
 - `Model`, [27](#)

- ModellImplementation, 33
- ExponencialFlow, 11
 - ExponencialFlow, 12
 - expression, 13
- ExponencialTest
 - FunctionalTests.cpp, 70
 - FunctionalTests.h, 72
- expression
 - ComplexFlow, 11
 - ExponencialFlow, 13
 - Flow, 14
 - FlowImplementation, 19
 - LogisticalFlow, 24
 - UnitTestFlow, 43
 - UnitTestFlow2, 45
- Flow, 13
 - ~Flow, 14
 - expression, 14
 - getName, 14
 - getSystemBegin, 14
 - getSystemEnd, 15
 - getValue, 15
 - setName, 15
 - setSystemBegin, 15
 - setSystemEnd, 17
 - setValue, 17
- FlowImplementation, 17
 - ~FlowImplementation, 19
 - expression, 19
 - FlowImplementation, 19
 - getName, 20
 - getSystemBegin, 20
 - getSystemEnd, 20
 - getValue, 20
 - name, 22
 - setName, 20
 - setSystemBegin, 21
 - setSystemEnd, 21
 - setValue, 21
 - systemBegin, 22
 - systemEnd, 22
 - value, 22
- flows
 - ModellImplementation, 35
- FunctionalTests.cpp
 - ComplexTest, 70
 - ExponencialTest, 70
 - LogisticalTest, 70
- FunctionalTests.h
 - ComplexTest, 72
 - ExponencialTest, 72
 - LogisticalTest, 72
- getFlowsIterator
 - Model, 27
 - ModellImplementation, 33
- getModelsIterator
 - Model, 27
- ModellImplementation, 33
- getName
 - Flow, 14
 - FlowImplementation, 20
 - Model, 28
 - ModellImplementation, 33
 - System, 37
 - SystemImplementation, 40
- getSystemBegin
 - Flow, 14
 - FlowImplementation, 20
- getSystemEnd
 - Flow, 15
 - FlowImplementation, 20
- getSystemsIterator
 - Model, 28
 - ModellImplementation, 34
- getTime
 - Model, 28
 - ModellImplementation, 34
- getValue
 - Flow, 15
 - FlowImplementation, 20
 - System, 37
 - SystemImplementation, 40
- HEX
 - CMakeCCompilerId.c, 48
 - CMakeCXXCompilerId.cpp, 51
- info_arch
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 52
- info_compiler
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 53
- info_language_extensions_default
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 53
- info_language_standard_default
 - CMakeCCompilerId.c, 50
 - CMakeCXXCompilerId.cpp, 53
- info_platform
 - CMakeCCompilerId.c, 50
 - CMakeCXXCompilerId.cpp, 53
- LogisticalFlow, 23
 - expression, 24
 - LogisticalFlow, 24
- LogisticalTest
 - FunctionalTests.cpp, 70
 - FunctionalTests.h, 72
- main
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 52
 - main.cpp, 67–69
- main.cpp
 - main, 67–69

- Model, 25
 - ~Model, 25
 - add, 26
 - createFlow, 26
 - createModel, 26
 - createSystem, 27
 - endFlows, 27
 - endModels, 27
 - endSystems, 27
 - getFlowsIterator, 27
 - getModelsIterator, 27
 - getName, 28
 - getSystemsIterator, 28
 - getTime, 28
 - setName, 28
 - setTime, 29
 - simulate, 29
- ModellImplementation, 29
 - ~ModellImplementation, 31
 - add, 32
 - createModel, 32
 - createSystem, 32
 - endFlows, 33
 - endModels, 33
 - endSystems, 33
 - flows, 35
 - getFlowsIterator, 33
 - getModelsIterator, 33
 - getName, 33
 - getSystemsIterator, 34
 - getTime, 34
 - ModellImplementation, 31
 - models, 35
 - name, 35
 - setName, 34
 - setTime, 34
 - simulate, 35
 - systems, 35
 - time, 36
- models
 - ModellImplementation, 35
- name
 - FlowImplementation, 22
 - ModellImplementation, 35
 - SystemImplementation, 41
- PLATFORM_ID
 - CMakeCCompilerId.c, 48
 - CMakeCXXCompilerId.cpp, 52
- README.md, 54
- runGlobal
 - unitTests.cpp, 91
 - unitTests.h, 92
- runUnitTestsFlow
 - unitFlow.cpp, 74
 - unitFlow.h, 77
- runUnitTestsModel
 - unitModel.cpp, 80
 - unitModel.h, 83
- runUnitTestsSystem
 - unitSystem.cpp, 86
 - unitSystem.h, 89
- setName
 - Flow, 15
 - FlowImplementation, 20
 - Model, 28
 - ModellImplementation, 34
 - System, 37
 - SystemImplementation, 40
- setSystemBegin
 - Flow, 15
 - FlowImplementation, 21
- setSystemEnd
 - Flow, 17
 - FlowImplementation, 21
- setTime
 - Model, 29
 - ModellImplementation, 34
- setValue
 - Flow, 17
 - FlowImplementation, 21
 - System, 37
 - SystemImplementation, 40
- simulate
 - Model, 29
 - ModellImplementation, 35
- src/lib/Flow.h, 54, 55
- src/lib/FlowImplementation.cpp, 56
- src/lib/FlowImplementation.h, 57
- src/lib/Model.h, 58, 59
- src/lib/ModellImplementation.cpp, 60
- src/lib/ModellImplementation.h, 62, 63
- src/lib/System.h, 64
- src/lib/SystemImplementation.cpp, 65
- src/lib/SystemImplementation.h, 65, 66
- src/main.cpp, 67
- STRINGIFY
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 52
- STRINGIFY_HELPER
 - CMakeCCompilerId.c, 49
 - CMakeCXXCompilerId.cpp, 52
- System, 36
 - ~System, 36
 - getName, 37
 - getValue, 37
 - setName, 37
 - setValue, 37
- systemBegin
 - FlowImplementation, 22
- systemEnd
 - FlowImplementation, 22
- SystemImplementation, 38
 - ~SystemImplementation, 39
 - getName, 40

- getValue, 40
 - name, 41
 - setName, 40
 - setValue, 40
 - SystemImplementation, 39
 - value, 41
- systems
 - ModellImplementation, 35
- test/functional/FunctionalTests.cpp, 70
- test/functional/FunctionalTests.h, 71, 72
- test/functional/main.cpp, 67
- test/unit/main.cpp, 68
- test/unit/unitFlow.cpp, 73
- test/unit/unitFlow.h, 76, 79
- test/unit/unitModel.cpp, 79
- test/unit/unitModel.h, 82, 85
- test/unit/unitSystem.cpp, 85
- test/unit/unitSystem.h, 88, 90
- test/unit/unitTests.cpp, 90
- test/unit/unitTests.h, 91, 92
- time
 - ModellImplementation, 36
- unitFlow.cpp
 - runUnitTestsFlow, 74
 - unitFlowDefaultConstructor, 74
 - unitFlowDestructor, 74
 - unitFlowExpression, 74
 - unitFlowGetName, 74
 - unitFlowGetSystemBegin, 74
 - unitFlowGetSystemEnd, 74
 - unitFlowGetValue, 75
 - unitFlowSetName, 75
 - unitFlowSetSystemBegin, 75
 - unitFlowSetSystemEnd, 75
 - unitFlowSetValue, 75
- unitFlow.h
 - runUnitTestsFlow, 77
 - unitFlowDefaultConstructor, 77
 - unitFlowDestructor, 77
 - unitFlowExpression, 77
 - unitFlowGetName, 77
 - unitFlowGetSystemBegin, 77
 - unitFlowGetSystemEnd, 78
 - unitFlowGetValue, 78
 - unitFlowSetName, 78
 - unitFlowSetSystemBegin, 78
 - unitFlowSetSystemEnd, 78
 - unitFlowSetValue, 78
- unitFlowDefaultConstructor
 - unitFlow.cpp, 74
 - unitFlow.h, 77
- unitFlowDestructor
 - unitFlow.cpp, 74
 - unitFlow.h, 77
- unitFlowExpression
 - unitFlow.cpp, 74
 - unitFlow.h, 77
- unitFlowGetName
 - unitFlow.cpp, 74
 - unitFlow.h, 77
- unitFlowGetSystemBegin
 - unitFlow.cpp, 74
 - unitFlow.h, 77
- unitFlowGetSystemEnd
 - unitFlow.cpp, 74
 - unitFlow.h, 78
- unitFlowGetValue
 - unitFlow.cpp, 75
 - unitFlow.h, 78
- unitFlowSetName
 - unitFlow.cpp, 75
 - unitFlow.h, 78
- unitFlowSetSystemBegin
 - unitFlow.cpp, 75
 - unitFlow.h, 78
- unitFlowSetSystemEnd
 - unitFlow.cpp, 75
 - unitFlow.h, 78
- unitFlowSetValue
 - unitFlow.cpp, 75
 - unitFlow.h, 78
- unitModel.cpp
 - runUnitTestsModel, 80
 - unitModelAddFlow, 80
 - unitModelAddSystem, 80
 - unitModelCreateFlow, 80
 - unitModelCreateModel, 80
 - unitModelCreateSystem, 80
 - unitModelDefaultConstructor, 81
 - unitModelDestructor, 81
 - unitModelGetName, 81
 - unitModelGetTime, 81
 - unitModelSetName, 81
 - unitModelSetTime, 81
 - unitModelSimulate, 81
- unitModel.h
 - runUnitTestsModel, 83
 - unitModelAddFlow, 83
 - unitModelAddSystem, 83
 - unitModelCreateFlow, 83
 - unitModelCreateModel, 83
 - unitModelCreateSystem, 83
 - unitModelDefaultConstructor, 84
 - unitModelDestructor, 84
 - unitModelGetName, 84
 - unitModelGetTime, 84
 - unitModelSetName, 84
 - unitModelSetTime, 84
 - unitModelSimulate, 84
- unitModelAddFlow
 - unitModel.cpp, 80
 - unitModel.h, 83
- unitModelAddSystem
 - unitModel.cpp, 80
 - unitModel.h, 83

- unitModelCreateFlow
 - unitModel.cpp, [80](#)
 - unitModel.h, [83](#)
- unitModelCreateModel
 - unitModel.cpp, [80](#)
 - unitModel.h, [83](#)
- unitModelCreateSystem
 - unitModel.cpp, [80](#)
 - unitModel.h, [83](#)
- unitModelDefaultConstructor
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitModelDestructor
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitModelGetName
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitModelGetTime
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitModelSetName
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitModelSetTime
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitModelSimulate
 - unitModel.cpp, [81](#)
 - unitModel.h, [84](#)
- unitSystem.cpp
 - runUnitTestsSystem, [86](#)
 - unitSystemDefaultConstructor, [86](#)
 - unitSystemDestructor, [87](#)
 - unitSystemGetName, [87](#)
 - unitSystemGetValue, [87](#)
 - unitSystemSetName, [87](#)
 - unitSystemSetValue, [87](#)
- unitSystem.h
 - runUnitTestsSystem, [89](#)
 - unitSystemDefaultConstructor, [89](#)
 - unitSystemDestructor, [89](#)
 - unitSystemGetName, [89](#)
 - unitSystemGetValue, [89](#)
 - unitSystemSetName, [89](#)
 - unitSystemSetValue, [89](#)
- unitSystemDefaultConstructor
 - unitSystem.cpp, [86](#)
 - unitSystem.h, [89](#)
- unitSystemDestructor
 - unitSystem.cpp, [87](#)
 - unitSystem.h, [89](#)
- unitSystemGetName
 - unitSystem.cpp, [87](#)
 - unitSystem.h, [89](#)
- unitSystemGetValue
 - unitSystem.cpp, [87](#)
 - unitSystem.h, [89](#)
- unitSystemSetName
 - unitSystem.cpp, [87](#)
 - unitSystem.h, [89](#)
- unitSystemSetValue
 - unitSystem.cpp, [87](#)
 - unitSystem.h, [89](#)
- UnitTestFlow, [41](#)
 - expression, [43](#)
 - UnitTestFlow, [42](#)
- UnitTestFlow2, [43](#)
 - expression, [45](#)
 - UnitTestFlow2, [44](#)
- unitTests.cpp
 - runGlobal, [91](#)
- unitTests.h
 - runGlobal, [92](#)
- value
 - FlowImplementation, [22](#)
 - SystemImplementation, [41](#)