

MyVensim

Generated by Doxygen 1.9.4



<b>1 Eng1</b>	<b>1</b>
<b>2 Hierarchical Index</b>	<b>3</b>
2.1 Class Hierarchy	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List	5
<b>4 File Index</b>	<b>7</b>
4.1 File List	7
<b>5 Class Documentation</b>	<b>9</b>
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 operator=()	20
5.4.3.7 setName()	21
5.4.3.8 setSystemBegin()	21
5.4.3.9 setSystemEnd()	21
5.4.3.10 setValue()	22
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	23
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	25
5.5.3.1 expression()	25
5.6 Model Class Reference	25
5.6.1 Constructor & Destructor Documentation	26
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 getFlowsIterator()	27
5.6.2.4 getName()	27
5.6.2.5 getSystemsIterator()	27
5.6.2.6 getTime()	27
5.6.2.7 setName()	27
5.6.2.8 setTime()	28
5.6.2.9 simulate()	28
5.7 ModelImplementation Class Reference	28
5.7.1 Detailed Description	30
5.7.2 Constructor & Destructor Documentation	30
5.7.2.1 ~ModelImplementation()	30
5.7.2.2 ModelImplementation()	30
5.7.3 Member Function Documentation	30
5.7.3.1 add() [1/2]	30
5.7.3.2 add() [2/2]	31
5.7.3.3 getFlowsIterator()	31

5.7.3.4 getName()	31
5.7.3.5 getSystemsIterator()	31
5.7.3.6 getTime()	32
5.7.3.7 operator=()	32
5.7.3.8 setName()	32
5.7.3.9 setTime()	32
5.7.3.10 simulate()	33
5.7.4 Member Data Documentation	33
5.7.4.1 flows	33
5.7.4.2 name	33
5.7.4.3 systems	33
5.7.4.4 time	34
5.8 System Class Reference	34
5.8.1 Constructor & Destructor Documentation	34
5.8.1.1 ~System()	34
5.8.2 Member Function Documentation	35
5.8.2.1 getName()	35
5.8.2.2 getValue()	35
5.8.2.3 setName()	35
5.8.2.4 setValue()	35
5.9 SystemImplementation Class Reference	36
5.9.1 Detailed Description	37
5.9.2 Constructor & Destructor Documentation	37
5.9.2.1 ~SystemImplementation()	37
5.9.2.2 SystemImplementation()	37
5.9.3 Member Function Documentation	38
5.9.3.1 getName()	38
5.9.3.2 getValue()	38
5.9.3.3 operator=()	38
5.9.3.4 setName()	38
5.9.3.5 setValue()	39
5.9.4 Member Data Documentation	39
5.9.4.1 name	39
5.9.4.2 value	39
5.10 UnitTestFlow Class Reference	40
5.10.1 Detailed Description	41
5.10.2 Constructor & Destructor Documentation	41
5.10.2.1 UnitTestFlow()	41
5.10.3 Member Function Documentation	41
5.10.3.1 expression()	41
5.11 UnitTestFlow2 Class Reference	42
5.11.1 Detailed Description	43

5.11.2 Constructor & Destructor Documentation	43
5.11.2.1 UnitTestFlow2()	43
5.11.3 Member Function Documentation	43
5.11.3.1 expression()	43
<b>6 File Documentation</b>	<b>45</b>
6.1 cmake-build-debug/CMakeCache.txt File Reference	45
6.2 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c File Reference	45
6.2.1 Macro Definition Documentation	45
6.2.1.1 __has_include	46
6.2.1.2 ARCHITECTURE_ID	46
6.2.1.3 C_VERSION	46
6.2.1.4 COMPILER_ID	46
6.2.1.5 DEC	46
6.2.1.6 HEX	46
6.2.1.7 PLATFORM_ID	47
6.2.1.8 STRINGIFY	47
6.2.1.9 STRINGIFY_HELPER	47
6.2.2 Function Documentation	47
6.2.2.1 main()	47
6.2.3 Variable Documentation	47
6.2.3.1 info_arch	47
6.2.3.2 info_compiler	47
6.2.3.3 info_language_extensions_default	48
6.2.3.4 info_language_standard_default	48
6.2.3.5 info_platform	48
6.3 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp File Reference	48
6.3.1 Macro Definition Documentation	49
6.3.1.1 __has_include	49
6.3.1.2 ARCHITECTURE_ID	49
6.3.1.3 COMPILER_ID	49
6.3.1.4 CXX_STD	49
6.3.1.5 DEC	49
6.3.1.6 HEX	50
6.3.1.7 PLATFORM_ID	50
6.3.1.8 STRINGIFY	50
6.3.1.9 STRINGIFY_HELPER	50
6.3.2 Function Documentation	50
6.3.2.1 main()	50
6.3.3 Variable Documentation	50
6.3.3.1 info_arch	51
6.3.3.2 info_compiler	51

6.3.3.3 info_language_extensions_default . . . . .	51
6.3.3.4 info_language_standard_default . . . . .	51
6.3.3.5 info_platform . . . . .	51
6.4 cmake-build-debug/CMakeFiles/clion-environment.txt File Reference . . . . .	52
6.5 cmake-build-debug/CMakeFiles/clion-log.txt File Reference . . . . .	52
6.6 cmake-build-debug/CMakeFiles/TargetDirectories.txt File Reference . . . . .	52
6.7 CMakeLists.txt File Reference . . . . .	52
6.8 README.md File Reference . . . . .	52
6.9 src/lib/Flow.h File Reference . . . . .	52
6.10 Flow.h . . . . .	53
6.11 src/lib/FlowImplementation.cpp File Reference . . . . .	54
6.12 src/lib/FlowImplementation.h File Reference . . . . .	55
6.13 FlowImplementation.h . . . . .	55
6.14 src/lib/Model.h File Reference . . . . .	56
6.15 Model.h . . . . .	57
6.16 src/lib/ModelImplementation.cpp File Reference . . . . .	58
6.17 src/lib/ModelImplementation.h File Reference . . . . .	59
6.18 ModelImplementation.h . . . . .	60
6.19 src/lib/System.h File Reference . . . . .	60
6.20 System.h . . . . .	61
6.21 src/lib/SystemImplementation.cpp File Reference . . . . .	62
6.22 src/lib/SystemImplementation.h File Reference . . . . .	62
6.23 SystemImplementation.h . . . . .	63
6.24 src/main.cpp File Reference . . . . .	64
6.24.1 Function Documentation . . . . .	64
6.24.1.1 main() . . . . .	64
6.25 test/functional/main.cpp File Reference . . . . .	64
6.25.1 Function Documentation . . . . .	65
6.25.1.1 main() . . . . .	65
6.26 test/unit/main.cpp File Reference . . . . .	65
6.26.1 Function Documentation . . . . .	66
6.26.1.1 main() . . . . .	66
6.27 test/functional/FunctionalTests.cpp File Reference . . . . .	67
6.27.1 Function Documentation . . . . .	67
6.27.1.1 ComplexTest() . . . . .	67
6.27.1.2 ExponencialTest() . . . . .	68
6.27.1.3 LogisticalTest() . . . . .	68
6.28 test/functional/FunctionalTests.h File Reference . . . . .	68
6.28.1 Function Documentation . . . . .	69
6.28.1.1 ComplexTest() . . . . .	69
6.28.1.2 ExponencialTest() . . . . .	69
6.28.1.3 LogisticalTest() . . . . .	69

6.29 FunctionalTests.h . . . . .	70
6.30 test/unit/unitFlow.cpp File Reference . . . . .	70
6.30.1 Function Documentation . . . . .	71
6.30.1.1 runUnitTestsFlow() . . . . .	71
6.30.1.2 unitFlowAssignmentOperator() . . . . .	71
6.30.1.3 unitFlowDefaultConstructor() . . . . .	71
6.30.1.4 unitFlowDestructor() . . . . .	71
6.30.1.5 unitFlowExpression() . . . . .	71
6.30.1.6 unitFlowGetName() . . . . .	72
6.30.1.7 unitFlowGetSystemBegin() . . . . .	72
6.30.1.8 unitFlowGetSystemEnd() . . . . .	72
6.30.1.9 unitFlowGetValue() . . . . .	72
6.30.1.10 unitFlowSetName() . . . . .	72
6.30.1.11 unitFlowSetSystemBegin() . . . . .	72
6.30.1.12 unitFlowSetSystemEnd() . . . . .	72
6.30.1.13 unitFlowSetValue() . . . . .	72
6.31 test/unit/unitFlow.h File Reference . . . . .	73
6.31.1 Function Documentation . . . . .	74
6.31.1.1 runUnitTestsFlow() . . . . .	74
6.31.1.2 unitFlowAssignmentOperator() . . . . .	74
6.31.1.3 unitFlowDefaultConstructor() . . . . .	74
6.31.1.4 unitFlowDestructor() . . . . .	74
6.31.1.5 unitFlowExpression() . . . . .	74
6.31.1.6 unitFlowGetName() . . . . .	75
6.31.1.7 unitFlowGetSystemBegin() . . . . .	75
6.31.1.8 unitFlowGetSystemEnd() . . . . .	75
6.31.1.9 unitFlowGetValue() . . . . .	75
6.31.1.10 unitFlowSetName() . . . . .	75
6.31.1.11 unitFlowSetSystemBegin() . . . . .	75
6.31.1.12 unitFlowSetSystemEnd() . . . . .	75
6.31.1.13 unitFlowSetValue() . . . . .	75
6.32 unitFlow.h . . . . .	76
6.33 test/unit/unitModel.cpp File Reference . . . . .	77
6.33.1 Function Documentation . . . . .	77
6.33.1.1 runUnitTestsModel() . . . . .	77
6.33.1.2 unitModeladdFlow() . . . . .	78
6.33.1.3 unitModeladdSystem() . . . . .	78
6.33.1.4 unitModelAssignmentOperator() . . . . .	78
6.33.1.5 unitModelDefaultConstructor() . . . . .	78
6.33.1.6 unitModelDestructor() . . . . .	78
6.33.1.7 unitModelGetName() . . . . .	78
6.33.1.8 unitModelGetTime() . . . . .	78



6.33.1.9 unitModelSetName()	78
6.33.1.10 unitModelSetTime()	79
6.33.1.11 unitModelSimulate()	79
6.34 test/unit/unitModel.h File Reference	79
6.34.1 Function Documentation	80
6.34.1.1 runUnitTestsModel()	80
6.34.1.2 unitModeladdFlow()	80
6.34.1.3 unitModeladdSystem()	81
6.34.1.4 unitModelAssignmentOperator()	81
6.34.1.5 unitModelDefaultConstructor()	81
6.34.1.6 unitModelDestructor()	81
6.34.1.7 unitModelGetName()	81
6.34.1.8 unitModelGetTime()	81
6.34.1.9 unitModelSetName()	81
6.34.1.10 unitModelSetTime()	81
6.34.1.11 unitModelSimulate()	82
6.35 unitModel.h	82
6.36 test/unit/unitSystem.cpp File Reference	83
6.36.1 Function Documentation	83
6.36.1.1 runUnitTestsSystem()	83
6.36.1.2 unitSystemAssignmentOperator()	84
6.36.1.3 unitSystemDefaultConstructor()	84
6.36.1.4 unitSystemDestructor()	84
6.36.1.5 unitSystemGetName()	84
6.36.1.6 unitSystemGetValue()	84
6.36.1.7 unitSystemSetName()	84
6.36.1.8 unitSystemSetValue()	84
6.37 test/unit/unitSystem.h File Reference	85
6.37.1 Function Documentation	86
6.37.1.1 runUnitTestsSystem()	86
6.37.1.2 unitSystemAssignmentOperator()	86
6.37.1.3 unitSystemDefaultConstructor()	86
6.37.1.4 unitSystemDestructor()	86
6.37.1.5 unitSystemGetName()	86
6.37.1.6 unitSystemGetValue()	86
6.37.1.7 unitSystemSetName()	87
6.37.1.8 unitSystemSetValue()	87
6.38 unitSystem.h	87
6.39 test/unit/unitTests.cpp File Reference	87
6.39.1 Function Documentation	88
6.39.1.1 runGlobal()	88
6.40 test/unit/unitTests.h File Reference	88

6.40.1 Function Documentation . . . . .	89
6.40.1.1 runGlobal() . . . . .	89
6.41 unitTests.h . . . . .	89
<b>Index</b>	<b>91</b>

# 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 . . . . .	40
UnitTestFlow2 . . . . .	42
Model . . . . .	25
ModelImplementation . . . . .	28
System . . . . .	34
SystemImplementation . . . . .	36



## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">ComplexFlow</a>	9
<a href="#">ExponencialFlow</a>	11
<a href="#">Flow</a>	13
<a href="#">FlowImplementation</a>	17
<a href="#">LogisticalFlow</a>	23
<a href="#">Model</a>	25
<a href="#">ModellImplementation</a>	28
<a href="#">System</a>	34
<a href="#">SystemImplementation</a>	36
<a href="#">UnitTestFlow</a>	40
<a href="#">UnitTestFlow2</a>	42





## 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	45
cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp	48
src/main.cpp	64
src/lib/Flow.h	52
src/lib/FlowImplementation.cpp	54
src/lib/FlowImplementation.h	55
src/lib/Model.h	56
src/lib/ModelImplementation.cpp	58
src/lib/ModelImplementation.h	59
src/lib/System.h	60
src/lib/SystemImplementation.cpp	62
src/lib/SystemImplementation.h	62
test/functional/FunctionalTests.cpp	67
test/functional/FunctionalTests.h	68
test/functional/main.cpp	64
test/unit/main.cpp	65
test/unit/unitFlow.cpp	70
test/unit/unitFlow.h	73
test/unit/unitModel.cpp	77
test/unit/unitModel.h	79
test/unit/unitSystem.cpp	83
test/unit/unitSystem.h	85
test/unit/unitTests.cpp	87
test/unit/unitTests.h	88

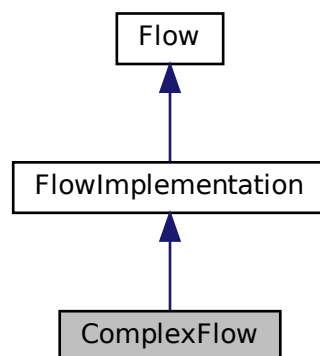


## Chapter 5

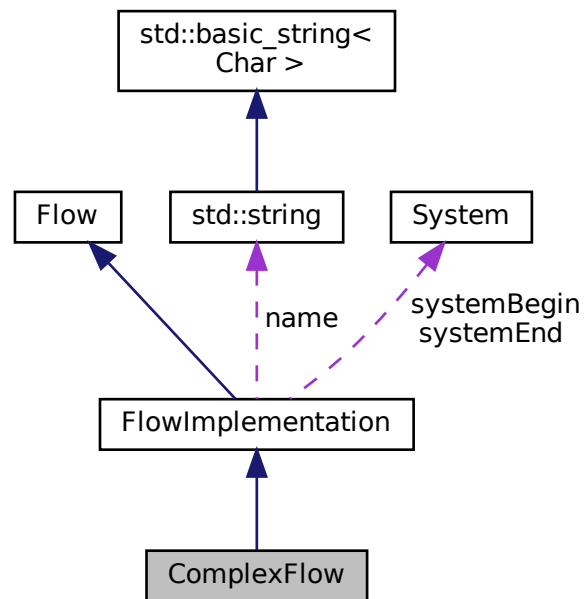
# Class Documentation

### 5.1 ComplexFlow Class Reference

Inheritance diagram for ComplexFlow:



Collaboration diagram for ComplexFlow:



## Public Member Functions

- `ComplexFlow` (`std::string name`, `double value`, `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,
    double value,
    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

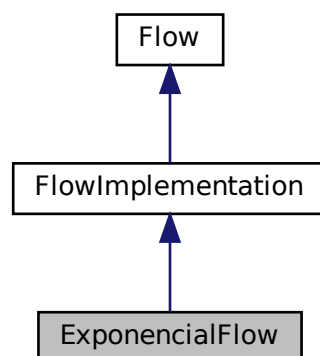
Implements [FlowImplementation](#).

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

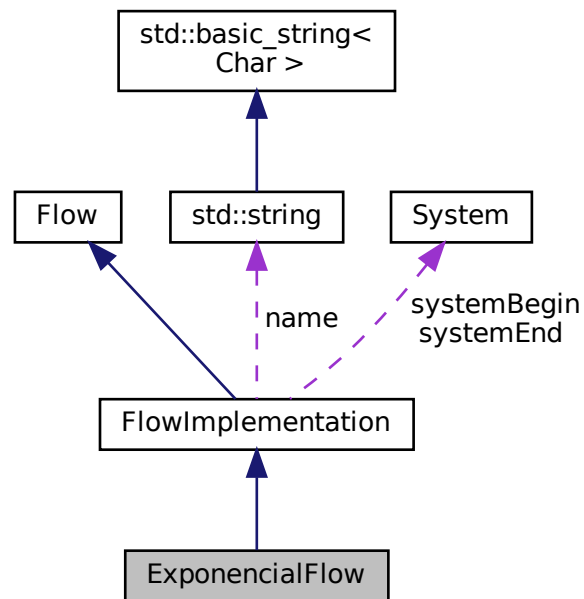
- test/functional/[FunctionalTests.cpp](#)

## 5.2 ExponentialFlow Class Reference

Inheritance diagram for ExponentialFlow:



Collaboration diagram for ExponencialFlow:



## Public Member Functions

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

```

ExponencialFlow::ExponencialFlow (
    std::string name,
    double value,
    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](#).

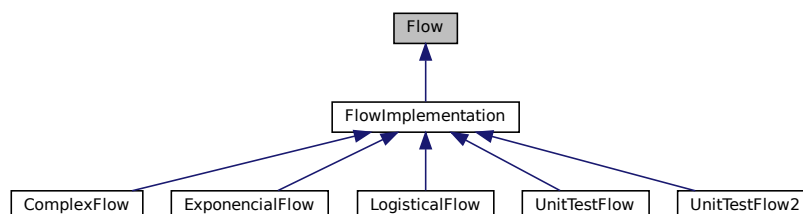
The documentation for this class was generated from the following file:

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

## 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](#).

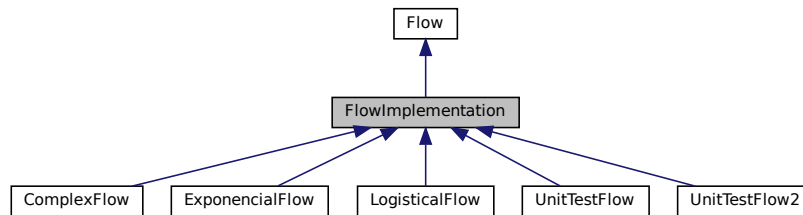
The documentation for this class was generated from the following file:

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

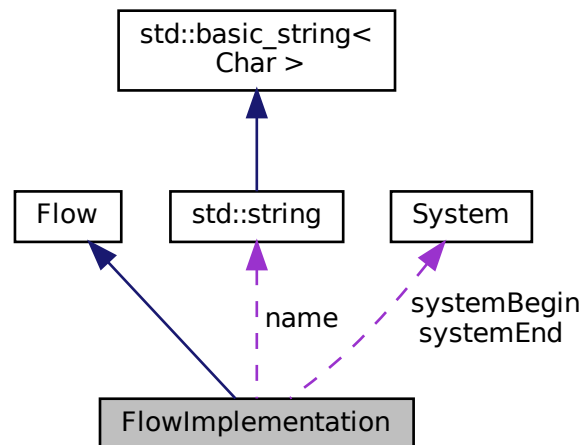
## 5.4 FlowImplementation Class Reference

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

Inheritance diagram for FlowImplementation:



Collaboration diagram for FlowImplementation:



## Public Member Functions

- `~FlowImplementation ()` override
- `FlowImplementation (std::string name, double value, System *systemBegin, System *systemEnd)`
- `FlowImplementation &operator= (const FlowImplementation &flow)`
- `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,
    double value,
    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 operator=()

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

Copy Assignment Operator

**Parameters**

<i>flow</i>	Flow to copy from
-------------	-------------------

**Returns**

Copied flow

**5.4.3.7 setName()**

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

Set system name

**Parameters**

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

Implements [Flow](#).

**5.4.3.8 setSystemBegin()**

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

Set systemBegin

**Parameters**

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

Implements [Flow](#).

**5.4.3.9 setSystemEnd()**

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

Set systemBegin

**Parameters**

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

Implements [Flow](#).

**5.4.3.10 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

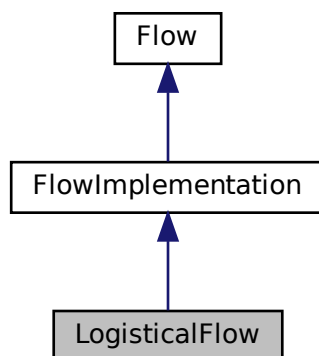
```
double FlowImplementation::value [protected]
```

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

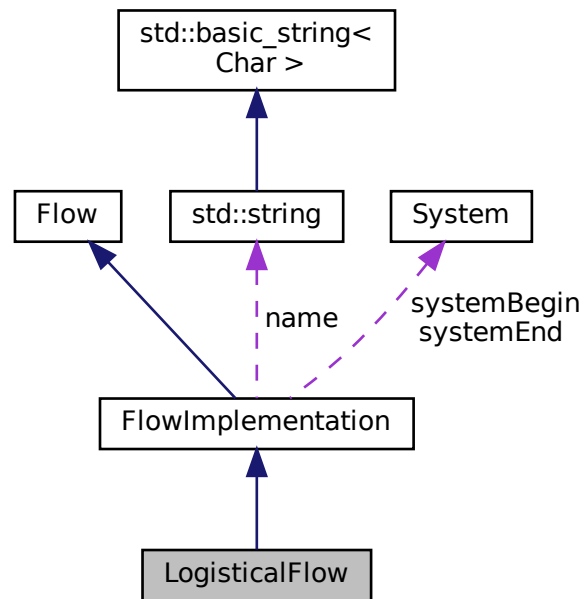
- [src/lib/FlowImplementation.h](#)
- [src/lib/FlowImplementation.cpp](#)

## 5.5 LogisticalFlow Class Reference

Inheritance diagram for LogisticalFlow:



Collaboration diagram for LogisticalFlow:



## Public Member Functions

- `LogisticalFlow` (`std::string name`, `double value`, `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 `value` minus the end system divided by seventy

### 5.5.2 Constructor & Destructor Documentation

#### 5.5.2.1 LogisticalFlow()

```

LogisticalFlow::LogisticalFlow (
    std::string name,
    double value,
    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](#).

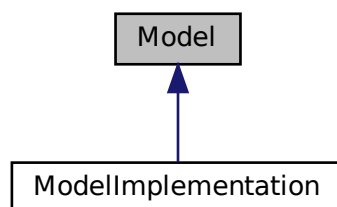
The documentation for this class was generated from the following file:

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

## 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

## 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>	<a href="#">Flow</a> 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>	<a href="#">System</a> to be added to the model
---------------	---

Implemented in [ModelImplementation](#).

### 5.6.2.3 getFlowsIterator()

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

Get model flows iterator

Implemented in [ModelImplementation](#).

### 5.6.2.4 getName()

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

Get model name

Implemented in [ModelImplementation](#).

### 5.6.2.5 getSystemsIterator()

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

Get model systems iterator

Implemented in [ModelImplementation](#).

### 5.6.2.6 getTime()

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

Get model time

Implemented in [ModelImplementation](#).

### 5.6.2.7 setName()

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

Set model name

## Parameters

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

Implemented in [ModellImplementation](#).

**5.6.2.8 setTime()**

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

Set model time

## Parameters

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

Implemented in [ModellImplementation](#).

**5.6.2.9 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 [ModellImplementation](#).

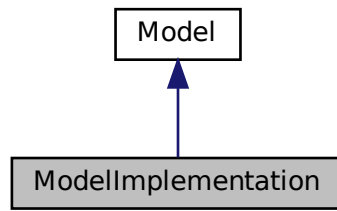
The documentation for this class was generated from the following file:

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

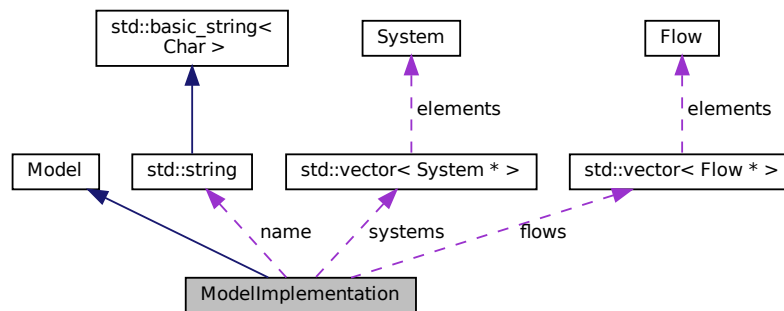
**5.7 ModellImplementation Class Reference**

```
#include <ModellImplementation.h>
```

Inheritance diagram for ModelImplementation:



Collaboration diagram for ModelImplementation:



## Public Member Functions

- `~ModelImplementation()` override
- `ModelImplementation(std::string name, double time)`
- `ModelImplementation & operator= (const ModelImplementation &model)`
- `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
- `void add(System *system)` override
- `void add(Flow *flow)` override
- `std::vector<System*>::iterator getSystemsIterator()` override
- `std::vector<Flow*>::iterator getFlowsIterator()` override

## Protected Attributes

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

### 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

#### 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], [virtual]
```

Add a flow to the model

##### Parameters

<i>flow</i>	<a href="#">Flow</a> to be added to the model
-------------	---



Implements [Model](#).

### 5.7.3.2 add() [2/2]

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

Add a system to the model

#### Parameters

<i>system</i>	<a href="#">System</a> to be added to the model
---------------	---

Implements [Model](#).

### 5.7.3.3 getFlowsIterator()

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

Get model flows iterator

Implements [Model](#).

### 5.7.3.4 getName()

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

Get model name

Implements [Model](#).

### 5.7.3.5 getSystemsIterator()

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

Get model systems iterator

Implements [Model](#).

#### 5.7.3.6 getTime()

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

Get model time

Implements [Model](#).

#### 5.7.3.7 operator=()

```
ModelImplementation & ModelImplementation::operator= (
    const ModelImplementation & model )
```

Copy Assignment Operator

Parameters

<i>model</i>	<a href="#">Model</a> to copy from
--------------	------------------------------------

Returns

Copied model

#### 5.7.3.8 setName()

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

Set model name

Parameters

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

Implements [Model](#).

#### 5.7.3.9 setTime()

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

Set model time

## Parameters

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

Implements [Model](#).

### 5.7.3.10 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 name

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

### 5.7.4.3 systems

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

#### 5.7.4.4 time

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

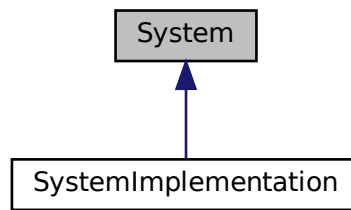
The documentation for this class was generated from the following files:

- [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**

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

Implemented in [SystemImplementation](#).

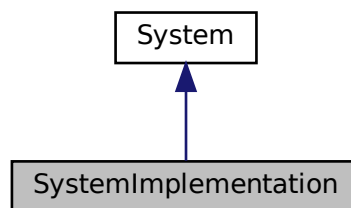
The documentation for this class was generated from the following file:

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

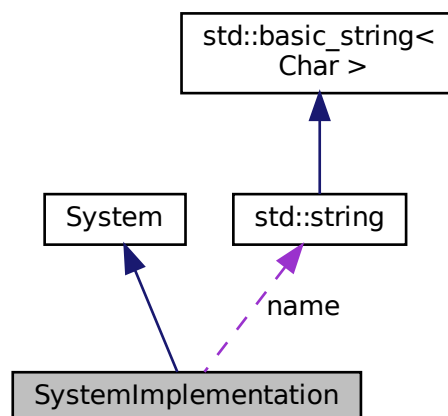
## 5.9 SystemImplementation Class Reference

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

Inheritance diagram for SystemImplementation:



Collaboration diagram for SystemImplementation:



## Public Member Functions

- [~SystemImplementation](#) () override
- [SystemImplementation](#) (std::string *name*, double *value*)
- [SystemImplementation](#) & [operator=](#) (const [SystemImplementation](#) &system)
- 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 operator=()

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

Copy Assignment Operator

#### Parameters

<i>system</i>	<a href="#">System</a> to copy from
---------------	-------------------------------------

#### Returns

Copied system

### 5.9.3.4 setName()

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

Set system name



**Parameters**

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

Implements [System](#).

**5.9.3.5 setValue()**

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

Set system value

**Parameters**

<i>v</i>	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]
```

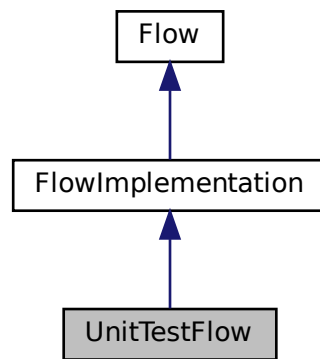
The documentation for this class was generated from the following files:

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

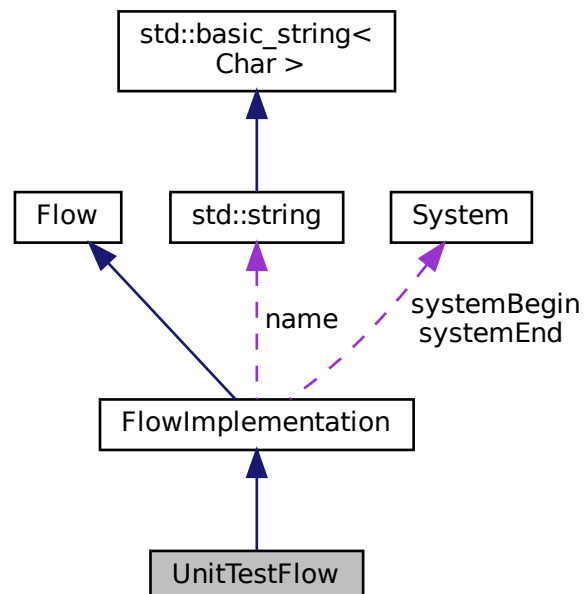
## 5.10 UnitTestFlow Class Reference

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

Inheritance diagram for UnitTestFlow:



Collaboration diagram for UnitTestFlow:



## Public Member Functions

- [UnitTestFlow](#) (std::string [name](#), double [value](#), [System](#) \*systemOut, [System](#) \*systemIn)
- 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 UnitTestFlow()

```
UnitTestFlow::UnitTestFlow (  
    std::string name,  
    double value,  
    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

[Flow](#) 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](#).

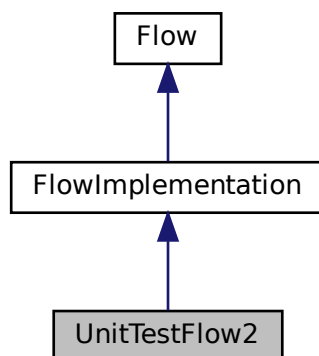
The documentation for this class was generated from the following file:

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

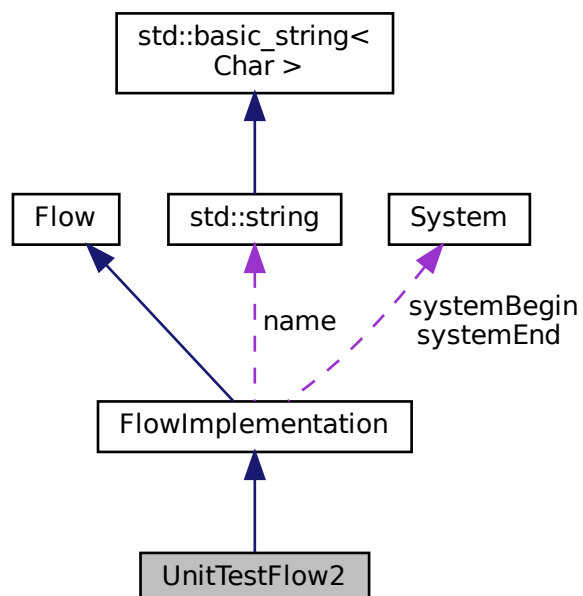
## 5.11 UnitTestFlow2 Class Reference

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

Inheritance diagram for UnitTestFlow2:



Collaboration diagram for UnitTestFlow2:



## Public Member Functions

- [UnitTestFlow2](#) (std::string *name*, double *value*, [System](#) \*systemOut, [System](#) \*systemIn)
- 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 UnitTestFlow2()

```
UnitTestFlow2::UnitTestFlow2 (
    std::string name,
    double value,
    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

[Flow](#) 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

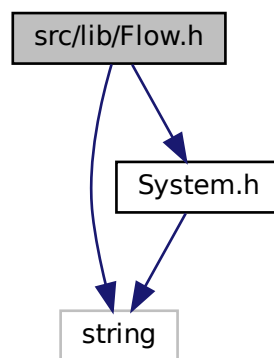
## 6.6 cmake-build-debug/CMakeFiles/TargetDirectories.txt File Reference

## 6.7 CMakeLists.txt File Reference

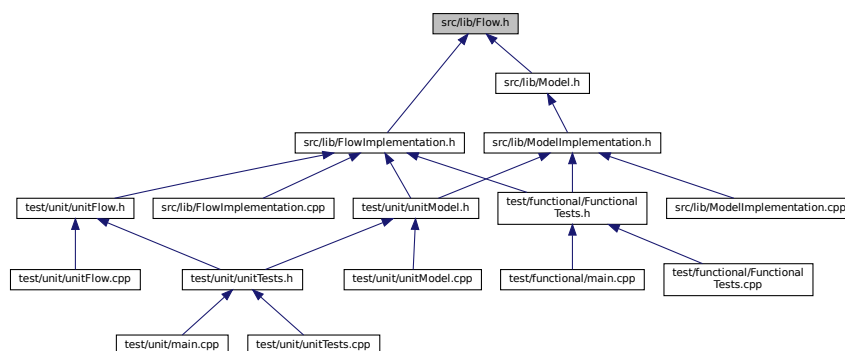
## 6.8 README.md File Reference

## 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

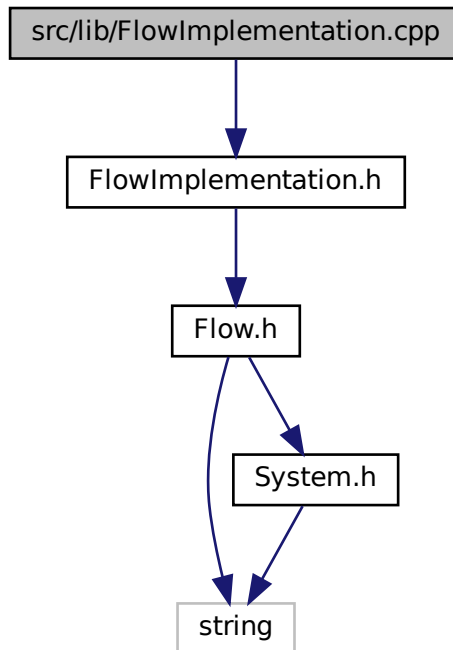
[Go to the documentation of this file.](#)

```
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:

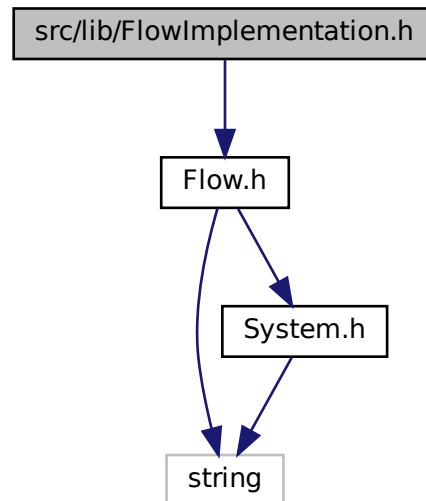




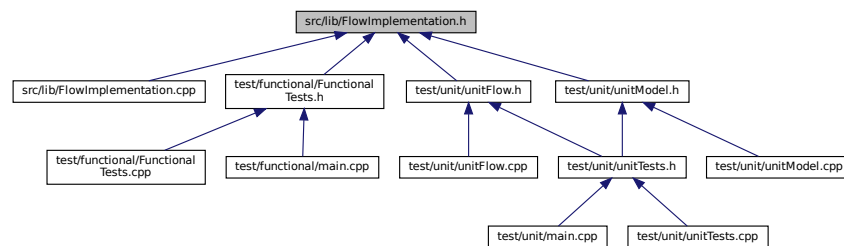
## 6.12 src/lib/FlowImplementation.h File Reference

```
#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 protected:
17     std::string name;
18     double value;
19     System* systemBegin;
20     System* systemEnd;
21
22 public:
23     ~FlowImplementation() override;
24
25     FlowImplementation(std::string name, double value, System* systemBegin, System* systemEnd);
26
27     FlowImplementation& operator=(const FlowImplementation& flow);
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

```

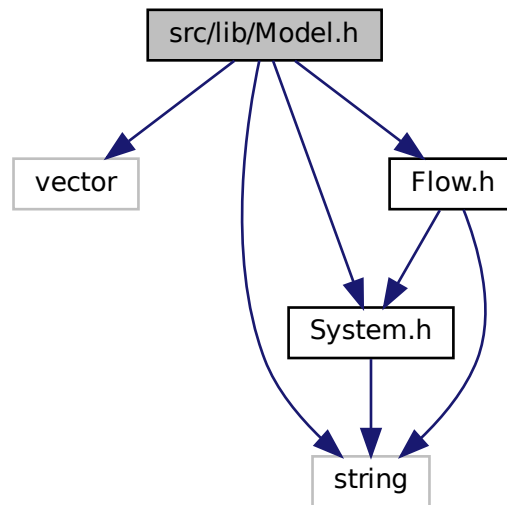
## 6.14 src/lib/Model.h File Reference

```

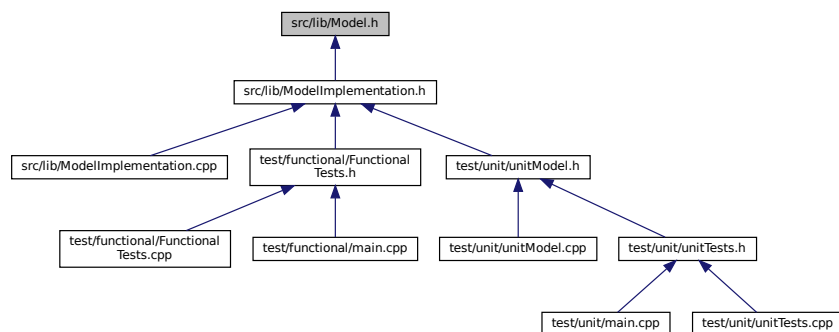
#include <vector>
#include <string>
#include "System.h"
#include "Flow.h"

```

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:
17     virtual ~Model() = default;
18     virtual void simulate(double, double, double) = 0;
19     virtual std::string getName() const = 0;
20     virtual void setName(std::string) = 0;
21     virtual double getTime() const = 0;
22     virtual void setTime(double) = 0;
23     virtual void add(System*) = 0;
24     virtual void add(Flow*) = 0;
25     virtual std::vector<System*>::iterator getSystemsIterator() = 0;
26     virtual std::vector<Flow*>::iterator getFlowsIterator() = 0;
27 };
28
29 #endif //ENGL_MODEL_H

```

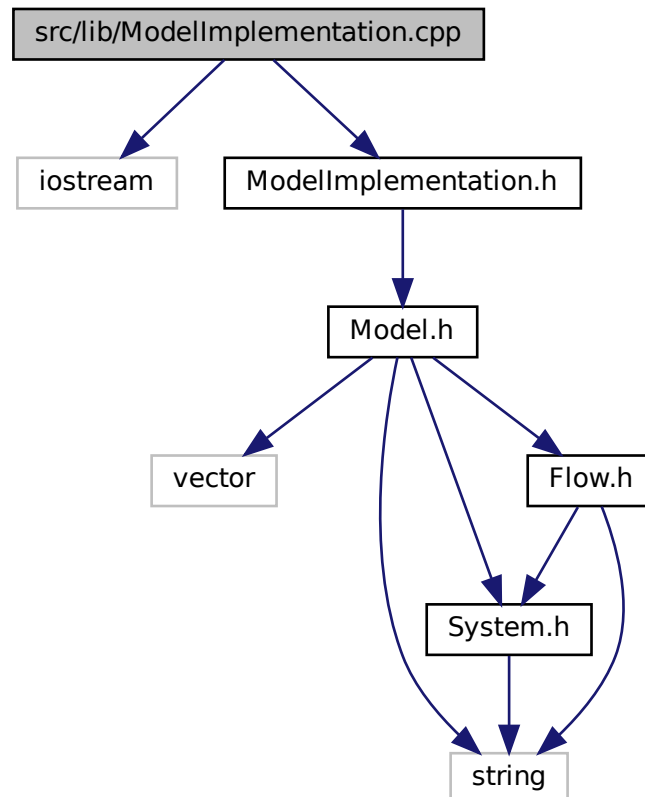
## 6.16 src/lib/ModelImplementation.cpp File Reference

```

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

```

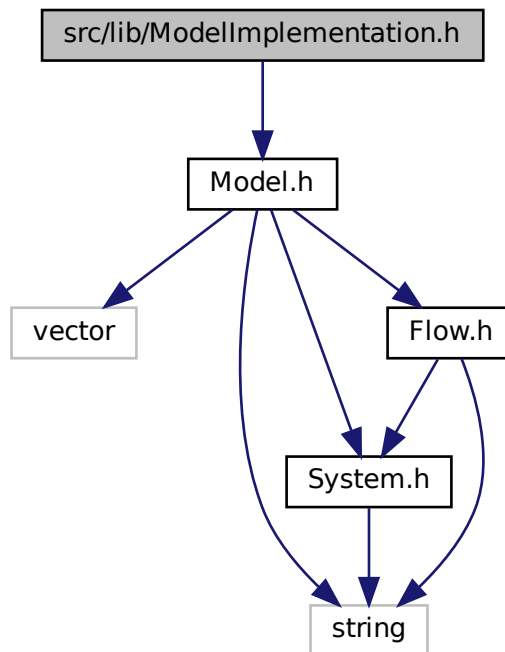
Include dependency graph for ModelImplementation.cpp:



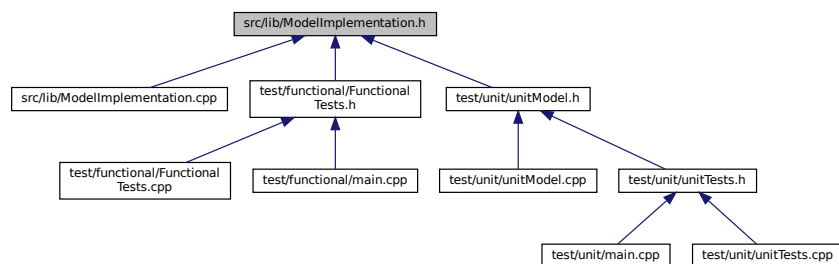
## 6.17 src/lib/ModellImplementation.h File Reference

```
#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.](#)

```

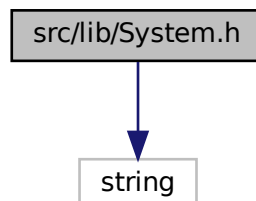
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENGL_MODELIMPLEMENTATION_H
6 #define ENGL_MODELIMPLEMENTATION_H
7
8
9 #include "Model.h"
10
11
12
13
14 class ModelImplementation: public Model {
15
16 private:
17     ModelImplementation(const ModelImplementation& model);
18
19 protected:
20     std::string name;
21     double time;
22     std::vector<System*> systems;
23     std::vector<Flow*> flows;
24
25 public:
26     ~ModelImplementation() override;
27
28     ModelImplementation(std::string name, double time);
29
30     ModelImplementation& operator=(const ModelImplementation& model);
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     void add(System* system) override;
43
44     void add(Flow* flow) override;
45
46     std::vector<System*>::iterator getSystemsIterator() override;
47
48     std::vector<Flow*>::iterator getFlowsIterator() override;
49 };
50
51 #endif //ENGL_MODELIMPLEMENTATION_H

```

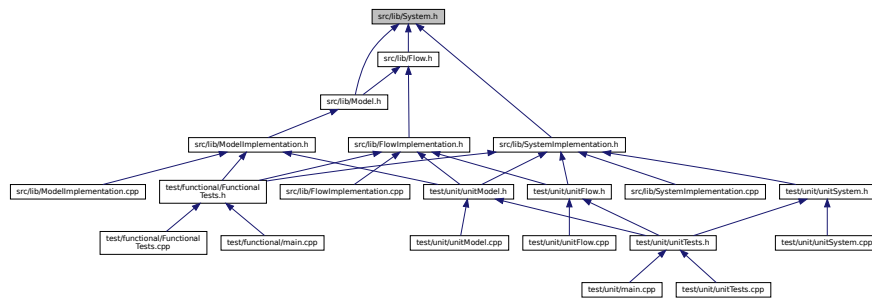
## 6.19 src/lib/System.h File Reference

```
#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.](#)

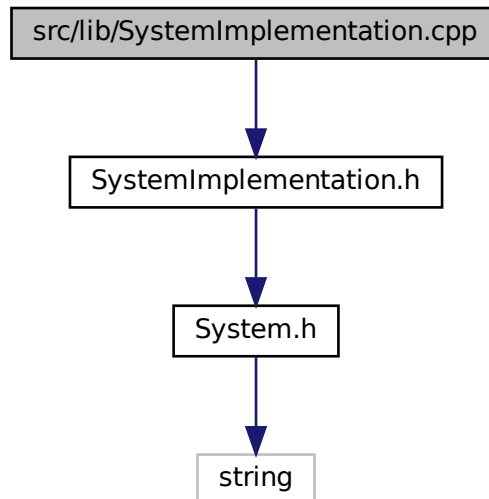
```

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"
```

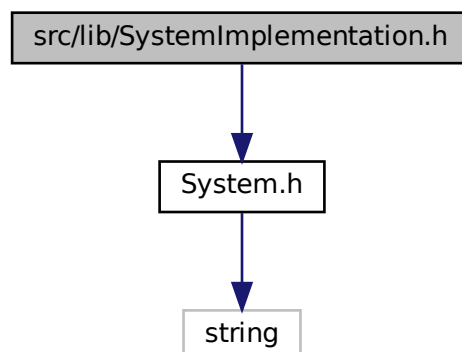
Include dependency graph for SystemImplementation.cpp:



## 6.22 src/lib/SystemImplementation.h File Reference

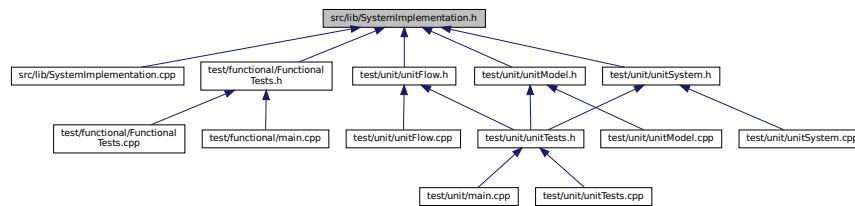
```
#include "System.h"
```

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.](#)

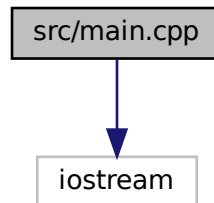
```

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
13 class SystemImplementation : public System {
14
15 private:
21     SystemImplementation(const SystemImplementation& system);
22
23 protected:
24     std::string name;
25     double value;
26
27 public:
31     ~SystemImplementation() override;
32
39     SystemImplementation(std::string name, double value);
40
46     SystemImplementation& operator=(const SystemImplementation& system);
47
51     std::string getName() const override;
52
57     void setName(std::string n) override;
58
62     double getValue() const override;
63
68     void setValue(double v) override;
69 };
70
71
72 #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()

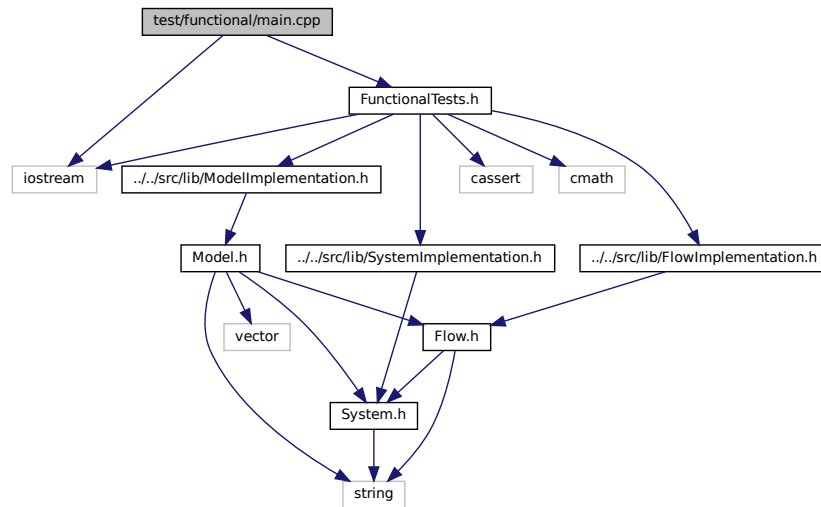
```
int main ( )
```

## 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

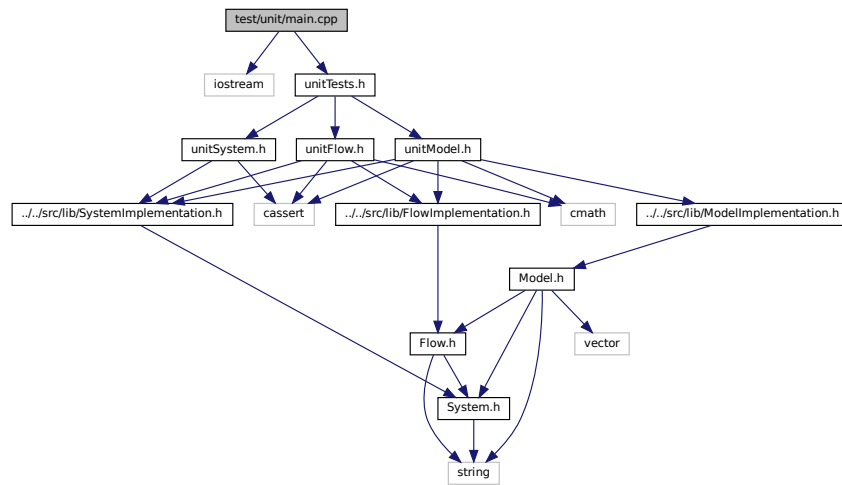
#### 6.25.1.1 `main()`

```
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

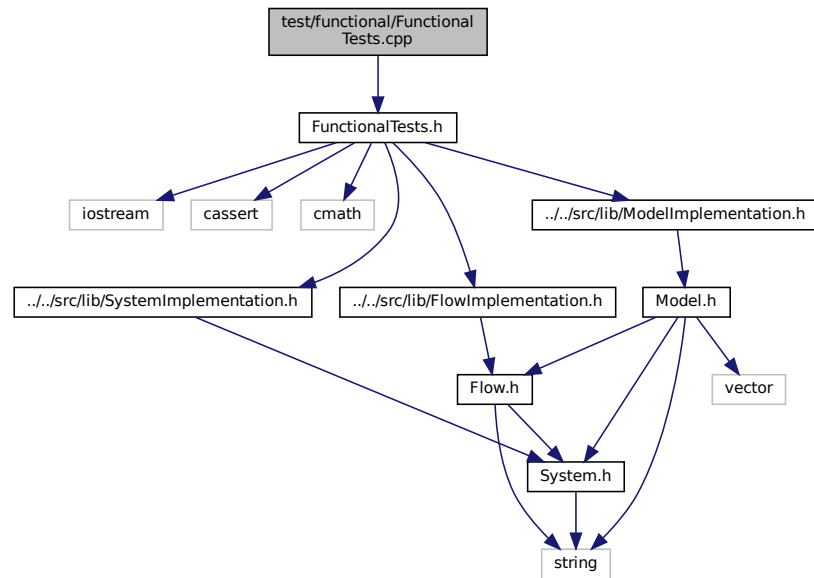
#### 6.26.1.1 main()

```
int main ( )
```

## 6.27 test/functional/FunctionalTests.cpp File Reference

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

Include dependency graph for FunctionalTests.cpp:



### Classes

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

### Functions

- void [ExponentialTest](#) ()
- 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 ExponentialTest()

```
void ExponentialTest ( )
```

Function to test the exponential flow.

### 6.27.1.3 LogisticalTest()

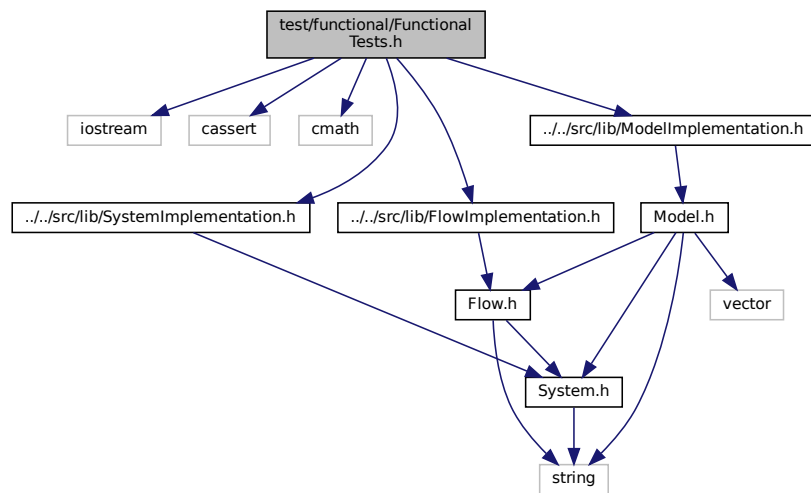
```
void LogisticalTest ( )
```

Function to test the logistical flow.

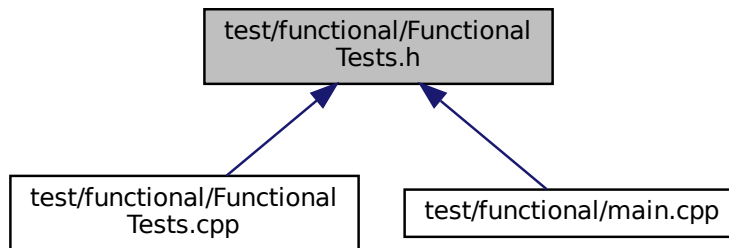
## 6.28 test/functional/FunctionalTests.h File Reference

```
#include <iostream>
#include <cassert>
#include <cmath>
#include "../src/lib/SystemImplementation.h"
#include "../src/lib/FlowImplementation.h"
#include "../src/lib/ModelImplementation.h"
```

Include dependency graph for FunctionalTests.h:



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



## 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 exponential 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.](#)

```

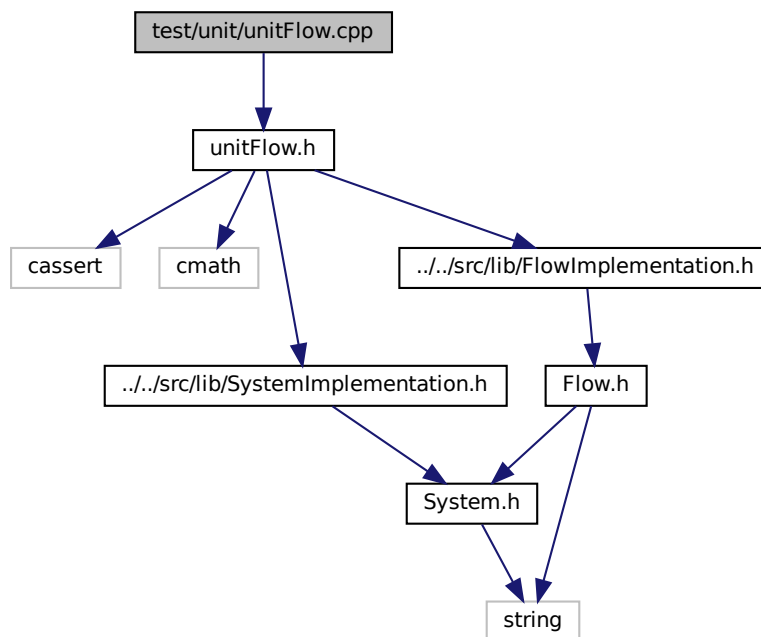
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENG1_FUNCTIONALTESTS_H
6 #define ENG1_FUNCTIONALTESTS_H
7
8 #include <iostream>
9 #include <cassert>
10 #include <cmath>
11
12 #include "../src/lib/SystemImplementation.h"
13 #include "../src/lib/FlowImplementation.h"
14 #include "../src/lib/ModelImplementation.h"
15
16 void ExponencialTest();
17 void ComplexTest();
18 void LogisticalTest();
19
20 #endif //ENG1_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 [unitFlowAssignmentOperator](#) ()
- 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 unitFlowAssignmentOperator()

```
void unitFlowAssignmentOperator ( )
```

Tests [Flow](#) assignment operator

### 6.30.1.3 unitFlowDefaultConstructor()

```
void unitFlowDefaultConstructor ( )
```

Tests [Flow](#) default constructor

### 6.30.1.4 unitFlowDestructor()

```
void unitFlowDestructor ( )
```

Tests [Flow](#) destructor

### 6.30.1.5 unitFlowExpression()

```
void unitFlowExpression ( )
```

Tests [Flow](#) expression

#### 6.30.1.6 unitFlowGetName()

```
void unitFlowGetName ( )
```

Tests [Flow](#) getName method

#### 6.30.1.7 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests [Flow](#) getSystemBegin method

#### 6.30.1.8 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests [Flow](#) getSystemEnd method

#### 6.30.1.9 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests [Flow](#) getValue method

#### 6.30.1.10 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests [Flow](#) setName method

#### 6.30.1.11 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests [Flow](#) setSystemBegin method

#### 6.30.1.12 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests [Flow](#) setSystemEnd method

#### 6.30.1.13 unitFlowSetValue()

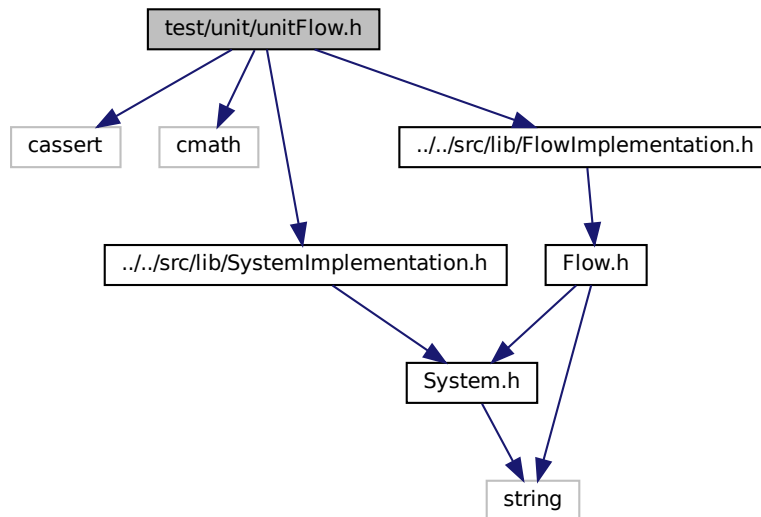
```
void unitFlowSetValue ( )
```

Tests [Flow](#) setValue method

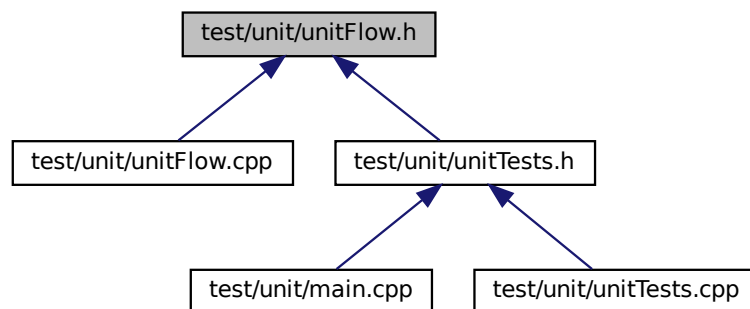
## 6.31 test/unit/unitFlow.h File Reference

```
#include <cassert>
#include <cmath>
#include "../src/lib/SystemImplementation.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 [unitFlowAssignmentOperator](#) ()
- 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 [unitFlowAssignmentOperator\(\)](#)

```
void unitFlowAssignmentOperator ( )
```

Tests [Flow](#) assignment operator

#### 6.31.1.3 [unitFlowDefaultConstructor\(\)](#)

```
void unitFlowDefaultConstructor ( )
```

Tests [Flow](#) default constructor

#### 6.31.1.4 [unitFlowDestructor\(\)](#)

```
void unitFlowDestructor ( )
```

Tests [Flow](#) destructor

#### 6.31.1.5 [unitFlowExpression\(\)](#)

```
void unitFlowExpression ( )
```

Tests [Flow](#) expression

#### 6.31.1.6 unitFlowGetName()

```
void unitFlowGetName ( )
```

Tests [Flow](#) getName method

#### 6.31.1.7 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests [Flow](#) getSystemBegin method

#### 6.31.1.8 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests [Flow](#) getSystemEnd method

#### 6.31.1.9 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests [Flow](#) getValue method

#### 6.31.1.10 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests [Flow](#) setName method

#### 6.31.1.11 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests [Flow](#) setSystemBegin method

#### 6.31.1.12 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests [Flow](#) setSystemEnd method

#### 6.31.1.13 unitFlowSetValue()

```
void unitFlowSetValue ( )
```

Tests [Flow](#) setValue method

## 6.32 unitFlow.h

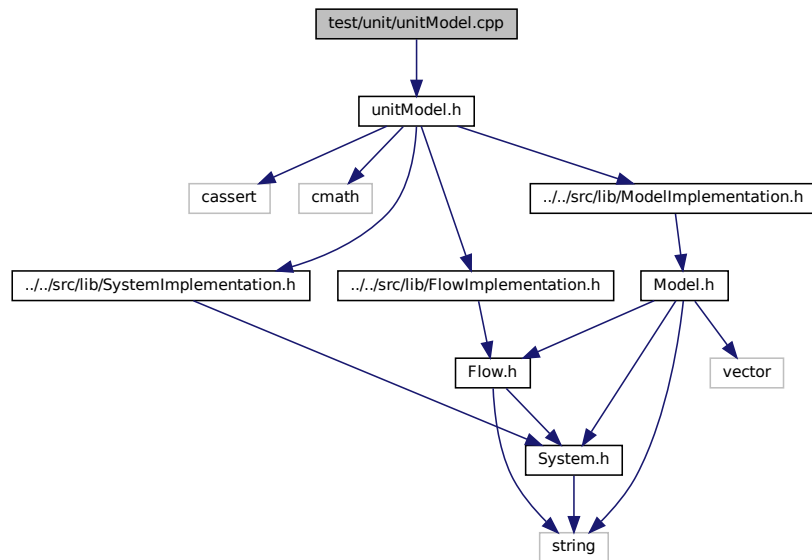
[Go to the documentation of this file.](#)

```
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/SystemImplementation.h"
13 #include "../src/lib/FlowImplementation.h"
14
15 class UnitTestFlow : public FlowImplementation{
16 public:
17     UnitTestFlow(std::string name, double value, System* systemOut, System* systemIn) :
18         FlowImplementation(name, value, systemOut, systemIn) {}
19     double expression() override {
20         return 0.01 * getSystemBegin()->getValue();
21     }
22 };
23
24 void unitFlowDestructor();
25 void unitFlowDefaultConstructor();
26 void unitFlowAssignmentOperator();
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 [unitModelAssignmentOperator](#) ()
- void [unitModelSimulate](#) ()
- void [unitModelGetName](#) ()
- void [unitModelSetName](#) ()
- void [unitModelGetTime](#) ()
- void [unitModelSetTime](#) ()
- void [unitModeladdSystem](#) ()
- void [unitModeladdFlow](#) ()
- 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 unitModelAssignmentOperator()

```
void unitModelAssignmentOperator ( )
```

Tests [Model](#) assignment operator

#### 6.33.1.5 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests [Model](#) default constructor

#### 6.33.1.6 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests [Model](#) destructor

#### 6.33.1.7 unitModelGetName()

```
void unitModelGetName ( )
```

Tests [Model](#) getName method

#### 6.33.1.8 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests [Model](#) getTime method

#### 6.33.1.9 unitModelSetName()

```
void unitModelSetName ( )
```

Tests [Model](#) setName method



## 6.33.1.10 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests [Model](#) setTime method

## 6.33.1.11 unitModelSimulate()

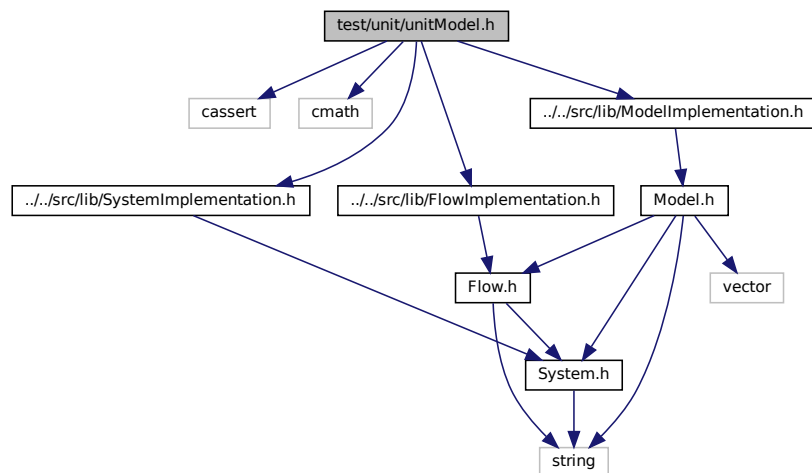
```
void unitModelSimulate ( )
```

Tests [Model](#) simulate method

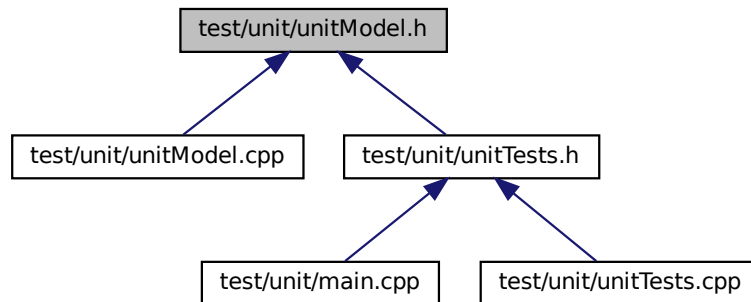
## 6.34 test/unit/unitModel.h File Reference

```
#include <cassert>
#include <cmath>
#include "../src/lib/SystemImplementation.h"
#include "../src/lib/FlowImplementation.h"
#include "../src/lib/ModelImplementation.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 [unitModelAssignmentOperator](#) ()
- void [unitModelSimulate](#) ()
- void [unitModelGetName](#) ()
- void [unitModelSetName](#) ()
- void [unitModelGetTime](#) ()
- void [unitModelSetTime](#) ()
- void [unitModeladdSystem](#) ()
- void [unitModeladdFlow](#) ()
- 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 unitModelAssignmentOperator()

```
void unitModelAssignmentOperator ( )
```

Tests [Model](#) assignment operator

#### 6.34.1.5 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests [Model](#) default constructor

#### 6.34.1.6 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests [Model](#) destructor

#### 6.34.1.7 unitModelGetName()

```
void unitModelGetName ( )
```

Tests [Model](#) getName method

#### 6.34.1.8 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests [Model](#) getTime method

#### 6.34.1.9 unitModelSetName()

```
void unitModelSetName ( )
```

Tests [Model](#) setName method

#### 6.34.1.10 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests [Model](#) setTime method

### 6.34.1.11 unitModelSimulate()

```
void unitModelSimulate ( )
```

Tests [Model](#) simulate method

## 6.35 unitModel.h

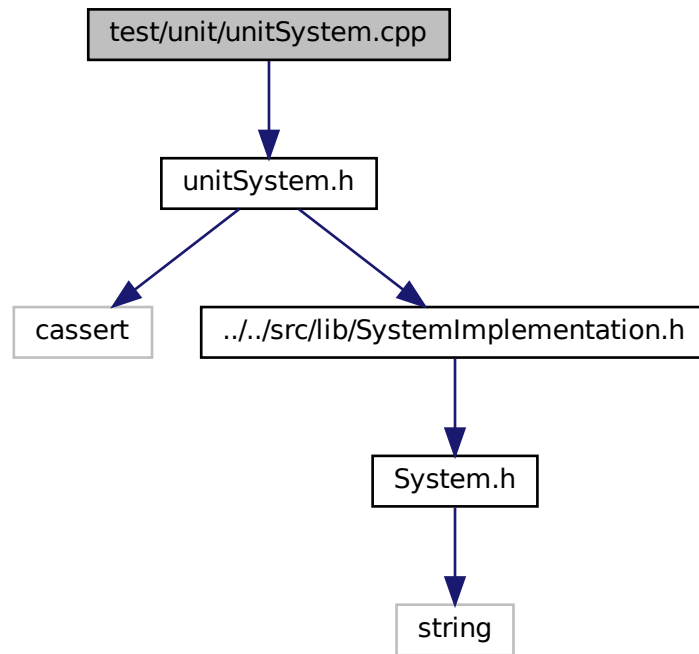
[Go to the documentation of this file.](#)

```
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/SystemImplementation.h"
13 #include "../src/lib/FlowImplementation.h"
14 #include "../src/lib/ModelImplementation.h"
15
16 class UnitTestFlow2 : public FlowImplementation{
17 public:
18     UnitTestFlow2(std::string name, double value, System* systemOut, System* systemIn) :
19         FlowImplementation(name, value, systemOut, systemIn) {}
20
21     double expression() override {
22         return 0.01 * getSystemBegin()->getValue();
23     }
24 };
25
26 void unitModelDestructor();
27 void unitModelDefaultConstructor();
28 void unitModelAssignmentOperator();
29 void unitModelSimulate();
30 void unitModelGetName();
31 void unitModelSetName();
32 void unitModelGetTime();
33 void unitModelSetTime();
34 void unitModeladdSystem();
35 void unitModeladdFlow();
36 void runUnitTestsModel();
37
38 #endif //ENGL_UNITMODEL_H
```

## 6.36 test/unit/unitSystem.cpp File Reference

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

Include dependency graph for unitSystem.cpp:



### Functions

- void [unitSystemDestructor](#) ()
- void [unitSystemDefaultConstructor](#) ()
- void [unitSystemAssignmentOperator](#) ()
- 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 unitSystemAssignmentOperator()

```
void unitSystemAssignmentOperator ( )
```

Tests [System](#) assignment operator

#### 6.36.1.3 unitSystemDefaultConstructor()

```
void unitSystemDefaultConstructor ( )
```

Tests [System](#) default constructor

#### 6.36.1.4 unitSystemDestructor()

```
void unitSystemDestructor ( )
```

Tests [System](#) destructor

#### 6.36.1.5 unitSystemGetName()

```
void unitSystemGetName ( )
```

Tests [System](#) getName method

#### 6.36.1.6 unitSystemGetValue()

```
void unitSystemGetValue ( )
```

Tests [System](#) getValue method

#### 6.36.1.7 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests [System](#) setName method

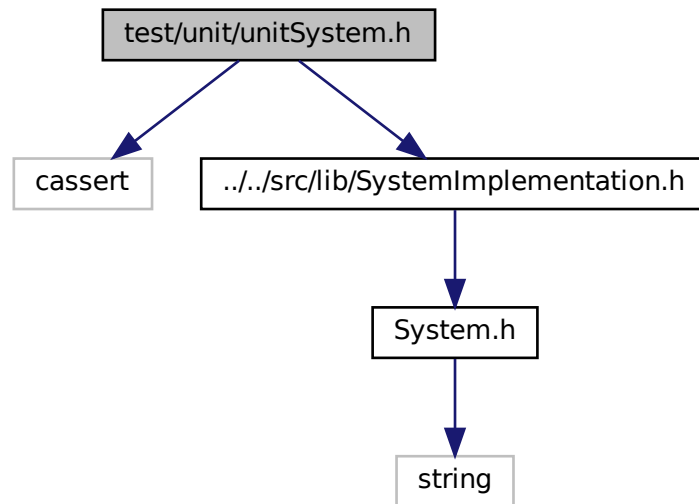
#### 6.36.1.8 unitSystemSetValue()

```
void unitSystemSetValue ( )
```

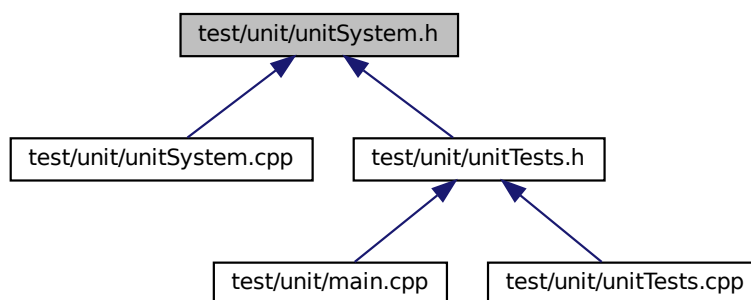
Tests [System](#) setValue method

## 6.37 test/unit/unitSystem.h File Reference

```
#include <cassert>
#include "../src/lib/SystemImplementation.h"
Include dependency graph for unitSystem.h:
```



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



### Functions

- void `unitSystemDestructor` ()
- void `unitSystemDefaultConstructor` ()
- void `unitSystemAssignmentOperator` ()
- 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 [unitSystemAssignmentOperator\(\)](#)

```
void unitSystemAssignmentOperator ( )
```

Tests [System](#) assignment operator

### 6.37.1.3 [unitSystemDefaultConstructor\(\)](#)

```
void unitSystemDefaultConstructor ( )
```

Tests [System](#) default constructor

### 6.37.1.4 [unitSystemDestructor\(\)](#)

```
void unitSystemDestructor ( )
```

Tests [System](#) destructor

### 6.37.1.5 [unitSystemGetName\(\)](#)

```
void unitSystemGetName ( )
```

Tests [System](#) getName method

### 6.37.1.6 [unitSystemGetValue\(\)](#)

```
void unitSystemGetValue ( )
```

Tests [System](#) getValue method



## 6.37.1.7 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests [System](#) setName method

## 6.37.1.8 unitSystemSetValue()

```
void unitSystemSetValue ( )
```

Tests [System](#) setValue method

## 6.38 unitSystem.h

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

```

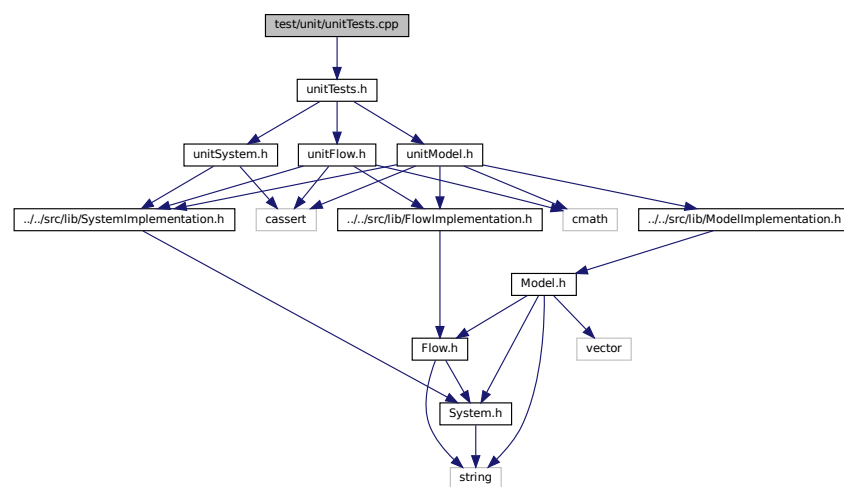
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENGL_UNITSYSTEM_H
6 #define ENGL_UNITSYSTEM_H
7
8 #include <cassert>
9
10 #include "../src/lib/SystemImplementation.h"
11
12 void unitSystemDestructor();
13 void unitSystemDefaultConstructor();
14 void unitSystemAssignmentOperator();
15 void unitSystemGetName();
16 void unitSystemSetName();
17 void unitSystemGetValue();
18 void unitSystemSetValue();
19 void runUnitTestsSystem();
20
21 #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

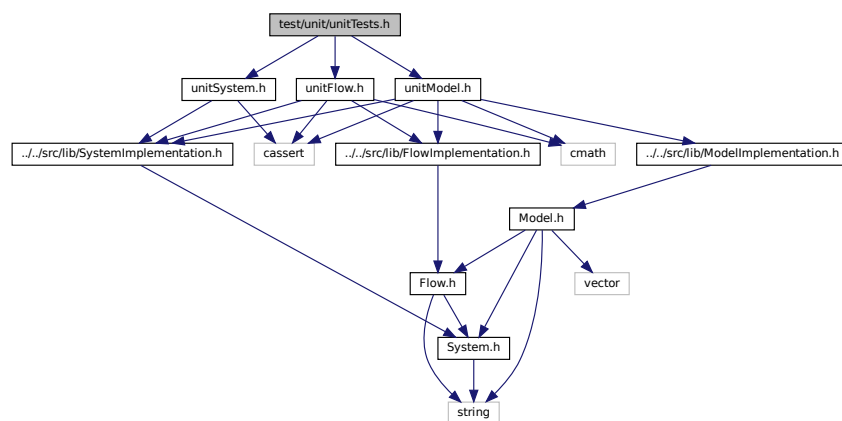
## 6.40 test/unit/unitTests.h File Reference

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

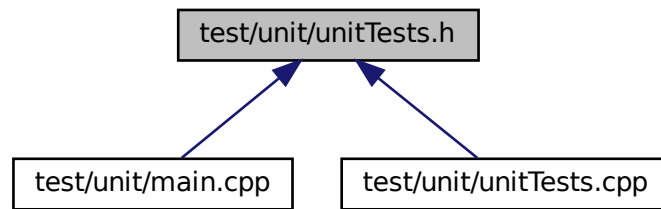
```
#include "unitFlow.h"
```

```
#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, [45](#)
    - CMakeCXXCompilerId.cpp, [49](#)
  - ~Flow
    - Flow, [14](#)
  - ~FlowImplementation
    - FlowImplementation, [19](#)
  - ~Model
    - Model, [26](#)
  - ~ModelImplementation
    - ModelImplementation, [30](#)
  - ~System
    - System, [34](#)
  - ~SystemImplementation
    - SystemImplementation, [37](#)
- add
  - Model, [26](#)
  - ModelImplementation, [30](#), [31](#)
- ARCHITECTURE\_ID
  - CMakeCCompilerId.c, [46](#)
  - CMakeCXXCompilerId.cpp, [49](#)
- C\_VERSION
  - CMakeCCompilerId.c, [46](#)
- cmake-build-debug/CMakeCache.txt, [45](#)
- cmake-build-debug/CMakeFiles/3.22.3/CompilerIdC/CMakeCCompilerId.c,  
[45](#)
- cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/CMakeCXXCompilerId.cpp,  
[48](#)
- cmake-build-debug/CMakeFiles/clion-environment.txt,  
[52](#)
- cmake-build-debug/CMakeFiles/clion-log.txt, [52](#)
- cmake-build-debug/CMakeFiles/TargetDirectories.txt,  
[52](#)
- CMakeCCompilerId.c
  - \_\_has\_include, [45](#)
  - ARCHITECTURE\_ID, [46](#)
  - C\_VERSION, [46](#)
  - COMPILER\_ID, [46](#)
  - DEC, [46](#)
  - HEX, [46](#)
  - info\_arch, [47](#)
  - info\_compiler, [47](#)
  - info\_language\_extensions\_default, [47](#)
  - info\_language\_standard\_default, [48](#)
  - info\_platform, [48](#)
  - main, [47](#)
  - PLATFORM\_ID, [46](#)
  - STRINGIFY, [47](#)
- STRINGIFY\_HELPER, [47](#)
- CMakeCXXCompilerId.cpp
  - \_\_has\_include, [49](#)
  - ARCHITECTURE\_ID, [49](#)
  - COMPILER\_ID, [49](#)
  - CXX\_STD, [49](#)
  - DEC, [49](#)
  - HEX, [49](#)
  - info\_arch, [50](#)
  - info\_compiler, [51](#)
  - info\_language\_extensions\_default, [51](#)
  - info\_language\_standard\_default, [51](#)
  - info\_platform, [51](#)
  - main, [50](#)
  - PLATFORM\_ID, [50](#)
  - STRINGIFY, [50](#)
  - STRINGIFY\_HELPER, [50](#)
- CMakeLists.txt, [52](#)
- COMPILER\_ID
  - CMakeCCompilerId.c, [46](#)
  - CMakeCXXCompilerId.cpp, [49](#)
- ComplexFlow, [9](#)
  - ComplexFlow, [10](#)
  - expression, [11](#)
- ComplexTest
  - FunctionalTests.cpp, [67](#)
  - FunctionalTests.h, [69](#)
- CXX\_STD
  - CMakeCXXCompilerId.cpp, [49](#)
- DEC
  - CMakeCCompilerId.c, [46](#)
  - CMakeCXXCompilerId.cpp, [49](#)
- ExponentialFlow, [11](#)
  - ExponentialFlow, [12](#)
  - expression, [13](#)
- ExponentialTest
  - FunctionalTests.cpp, [67](#)
  - FunctionalTests.h, [69](#)
- expression
  - ComplexFlow, [11](#)
  - ExponentialFlow, [13](#)
  - Flow, [14](#)
  - FlowImplementation, [19](#)
  - LogisticalFlow, [25](#)
  - UnitTestFlow, [41](#)
  - UnitTestFlow2, [43](#)
- 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](#)
  - operator=, [20](#)
  - setName, [21](#)
  - setSystemBegin, [21](#)
  - setSystemEnd, [21](#)
  - setValue, [22](#)
  - systemBegin, [22](#)
  - systemEnd, [22](#)
  - value, [22](#)
- flows
  - ModellImplementation, [33](#)
- FunctionalTests.cpp
  - ComplexTest, [67](#)
  - ExponencialTest, [67](#)
  - LogisticalTest, [68](#)
- FunctionalTests.h
  - ComplexTest, [69](#)
  - ExponencialTest, [69](#)
  - LogisticalTest, [69](#)
- getFlowsIterator
  - Model, [27](#)
  - ModellImplementation, [31](#)
- getName
  - Flow, [14](#)
  - FlowImplementation, [20](#)
  - Model, [27](#)
  - ModellImplementation, [31](#)
  - System, [35](#)
  - SystemImplementation, [38](#)
- getSystemBegin
  - Flow, [14](#)
  - FlowImplementation, [20](#)
- getSystemEnd
  - Flow, [15](#)
  - FlowImplementation, [20](#)
- getSystemsIterator
  - Model, [27](#)
  - ModellImplementation, [31](#)
- getTime
  - Model, [27](#)
  - ModellImplementation, [31](#)
- getValue
  - Flow, [15](#)
  - FlowImplementation, [20](#)
  - System, [35](#)
  - SystemImplementation, [38](#)
- HEX
  - CMakeCCompilerId.c, [46](#)
  - CMakeCXXCompilerId.cpp, [49](#)
- info\_arch
  - CMakeCCompilerId.c, [47](#)
  - CMakeCXXCompilerId.cpp, [50](#)
- info\_compiler
  - CMakeCCompilerId.c, [47](#)
  - CMakeCXXCompilerId.cpp, [51](#)
- info\_language\_extensions\_default
  - CMakeCCompilerId.c, [47](#)
  - CMakeCXXCompilerId.cpp, [51](#)
- info\_language\_standard\_default
  - CMakeCCompilerId.c, [48](#)
  - CMakeCXXCompilerId.cpp, [51](#)
- info\_platform
  - CMakeCCompilerId.c, [48](#)
  - CMakeCXXCompilerId.cpp, [51](#)
- LogisticalFlow, [23](#)
  - expression, [25](#)
  - LogisticalFlow, [24](#)
- LogisticalTest
  - FunctionalTests.cpp, [68](#)
  - FunctionalTests.h, [69](#)
- main
  - CMakeCCompilerId.c, [47](#)
  - CMakeCXXCompilerId.cpp, [50](#)
  - main.cpp, [64–66](#)
- main.cpp
  - main, [64–66](#)
- Model, [25](#)
  - ~Model, [26](#)
  - add, [26](#)
  - getFlowsIterator, [27](#)
  - getName, [27](#)
  - getSystemsIterator, [27](#)
  - getTime, [27](#)
  - setName, [27](#)
  - setTime, [28](#)
  - simulate, [28](#)
- ModellImplementation, [28](#)
  - ~ModellImplementation, [30](#)
  - add, [30, 31](#)
  - flows, [33](#)
  - getFlowsIterator, [31](#)
  - getName, [31](#)
  - getSystemsIterator, [31](#)
  - getTime, [31](#)
  - ModellImplementation, [30](#)

- name, 33
- operator=, 32
- setName, 32
- setTime, 32
- simulate, 33
- systems, 33
- time, 33
- name
  - FlowImplementation, 22
  - ModellImplementation, 33
  - SystemImplementation, 39
- operator=
  - FlowImplementation, 20
  - ModellImplementation, 32
  - SystemImplementation, 38
- PLATFORM\_ID
  - CMakeCCompilerId.c, 46
  - CMakeCXXCompilerId.cpp, 50
- README.md, 52
- runGlobal
  - unitTests.cpp, 88
  - unitTests.h, 89
- runUnitTestsFlow
  - unitFlow.cpp, 71
  - unitFlow.h, 74
- runUnitTestsModel
  - unitModel.cpp, 77
  - unitModel.h, 80
- runUnitTestsSystem
  - unitSystem.cpp, 83
  - unitSystem.h, 86
- setName
  - Flow, 15
  - FlowImplementation, 21
  - Model, 27
  - ModellImplementation, 32
  - System, 35
  - SystemImplementation, 38
- setSystemBegin
  - Flow, 15
  - FlowImplementation, 21
- setSystemEnd
  - Flow, 17
  - FlowImplementation, 21
- setTime
  - Model, 28
  - ModellImplementation, 32
- setValue
  - Flow, 17
  - FlowImplementation, 22
  - System, 35
  - SystemImplementation, 39
- simulate
  - Model, 28
  - ModellImplementation, 33
- src/lib/Flow.h, 52, 53
- src/lib/FlowImplementation.cpp, 54
- src/lib/FlowImplementation.h, 55
- src/lib/Model.h, 56, 57
- src/lib/ModellImplementation.cpp, 58
- src/lib/ModellImplementation.h, 59, 60
- src/lib/System.h, 60, 61
- src/lib/SystemImplementation.cpp, 62
- src/lib/SystemImplementation.h, 62, 63
- src/main.cpp, 64
- STRINGIFY
  - CMakeCCompilerId.c, 47
  - CMakeCXXCompilerId.cpp, 50
- STRINGIFY\_HELPER
  - CMakeCCompilerId.c, 47
  - CMakeCXXCompilerId.cpp, 50
- System, 34
  - ~System, 34
  - getName, 35
  - getValue, 35
  - setName, 35
  - setValue, 35
- systemBegin
  - FlowImplementation, 22
- systemEnd
  - FlowImplementation, 22
- SystemImplementation, 36
  - ~SystemImplementation, 37
  - getName, 38
  - getValue, 38
  - name, 39
  - operator=, 38
  - setName, 38
  - setValue, 39
  - SystemImplementation, 37
  - value, 39
- systems
  - ModellImplementation, 33
- test/functional/FunctionalTests.cpp, 67
- test/functional/FunctionalTests.h, 68, 70
- test/functional/main.cpp, 64
- test/unit/main.cpp, 65
- test/unit/unitFlow.cpp, 70
- test/unit/unitFlow.h, 73, 76
- test/unit/unitModel.cpp, 77
- test/unit/unitModel.h, 79, 82
- test/unit/unitSystem.cpp, 83
- test/unit/unitSystem.h, 85, 87
- test/unit/unitTests.cpp, 87
- test/unit/unitTests.h, 88, 89
- time
  - ModellImplementation, 33
- unitFlow.cpp
  - runUnitTestsFlow, 71
  - unitFlowAssignmentOperator, 71
  - unitFlowDefaultConstructor, 71

- unitFlowDestructor, 71
- unitFlowExpression, 71
- unitFlowGetName, 71
- unitFlowGetSystemBegin, 72
- unitFlowGetSystemEnd, 72
- unitFlowGetValue, 72
- unitFlowSetName, 72
- unitFlowSetSystemBegin, 72
- unitFlowSetSystemEnd, 72
- unitFlowSetValue, 72
- unitFlow.h
  - runUnitTestsFlow, 74
  - unitFlowAssignmentOperator, 74
  - unitFlowDefaultConstructor, 74
  - unitFlowDestructor, 74
  - unitFlowExpression, 74
  - unitFlowGetName, 74
  - unitFlowGetSystemBegin, 75
  - unitFlowGetSystemEnd, 75
  - unitFlowGetValue, 75
  - unitFlowSetName, 75
  - unitFlowSetSystemBegin, 75
  - unitFlowSetSystemEnd, 75
  - unitFlowSetValue, 75
- unitFlowAssignmentOperator
  - unitFlow.cpp, 71
  - unitFlow.h, 74
- unitFlowDefaultConstructor
  - unitFlow.cpp, 71
  - unitFlow.h, 74
- unitFlowDestructor
  - unitFlow.cpp, 71
  - unitFlow.h, 74
- unitFlowExpression
  - unitFlow.cpp, 71
  - unitFlow.h, 74
- unitFlowGetName
  - unitFlow.cpp, 71
  - unitFlow.h, 74
- unitFlowGetSystemBegin
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitFlowGetSystemEnd
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitFlowGetValue
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitFlowSetName
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitFlowSetSystemBegin
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitFlowSetSystemEnd
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitFlowSetValue
  - unitFlow.cpp, 72
  - unitFlow.h, 75
- unitModel.cpp
  - runUnitTestsModel, 77
  - unitModeladdFlow, 77
  - unitModeladdSystem, 78
  - unitModelAssignmentOperator, 78
  - unitModelDefaultConstructor, 78
  - unitModelDestructor, 78
  - unitModelGetName, 78
  - unitModelGetTime, 78
  - unitModelSetName, 78
  - unitModelSetTime, 78
  - unitModelSimulate, 79
- unitModel.h
  - runUnitTestsModel, 80
  - unitModeladdFlow, 80
  - unitModeladdSystem, 80
  - unitModelAssignmentOperator, 81
  - unitModelDefaultConstructor, 81
  - unitModelDestructor, 81
  - unitModelGetName, 81
  - unitModelGetTime, 81
  - unitModelSetName, 81
  - unitModelSetTime, 81
  - unitModelSimulate, 81
- unitModeladdFlow
  - unitModel.cpp, 77
  - unitModel.h, 80
- unitModeladdSystem
  - unitModel.cpp, 78
  - unitModel.h, 80
- unitModelAssignmentOperator
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelDefaultConstructor
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelDestructor
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelGetName
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelGetTime
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelSetName
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelSetTime
  - unitModel.cpp, 78
  - unitModel.h, 81
- unitModelSimulate
  - unitModel.cpp, 79
  - unitModel.h, 81
- unitSystem.cpp
  - runUnitTestsSystem, 83



- unitSystemAssignmentOperator, [83](#)
- unitSystemDefaultConstructor, [84](#)
- unitSystemDestructor, [84](#)
- unitSystemGetName, [84](#)
- unitSystemGetValue, [84](#)
- unitSystemSetName, [84](#)
- unitSystemSetValue, [84](#)
- unitSystem.h
  - runUnitTestsSystem, [86](#)
  - unitSystemAssignmentOperator, [86](#)
  - unitSystemDefaultConstructor, [86](#)
  - unitSystemDestructor, [86](#)
  - unitSystemGetName, [86](#)
  - unitSystemGetValue, [86](#)
  - unitSystemSetName, [86](#)
  - unitSystemSetValue, [87](#)
- unitSystemAssignmentOperator
  - unitSystem.cpp, [83](#)
  - unitSystem.h, [86](#)
- unitSystemDefaultConstructor
  - unitSystem.cpp, [84](#)
  - unitSystem.h, [86](#)
- unitSystemDestructor
  - unitSystem.cpp, [84](#)
  - unitSystem.h, [86](#)
- unitSystemGetName
  - unitSystem.cpp, [84](#)
  - unitSystem.h, [86](#)
- unitSystemGetValue
  - unitSystem.cpp, [84](#)
  - unitSystem.h, [86](#)
- unitSystemSetName
  - unitSystem.cpp, [84](#)
  - unitSystem.h, [86](#)
- unitSystemSetValue
  - unitSystem.cpp, [84](#)
  - unitSystem.h, [87](#)
- UnitTestFlow, [40](#)
  - expression, [41](#)
  - UnitTestFlow, [41](#)
- UnitTestFlow2, [42](#)
  - expression, [43](#)
  - UnitTestFlow2, [43](#)
- unitTests.cpp
  - runGlobal, [88](#)
- unitTests.h
  - runGlobal, [89](#)
- value
  - FlowImplementation, [22](#)
  - SystemImplementation, [39](#)