MyVensim

Generated by Doxygen 1.9.4

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:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

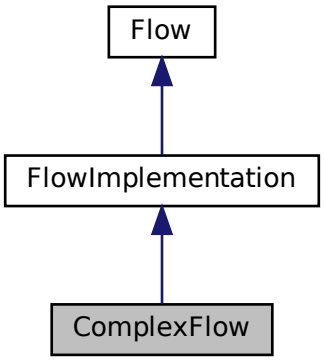
Chapter 5

Class Documentation

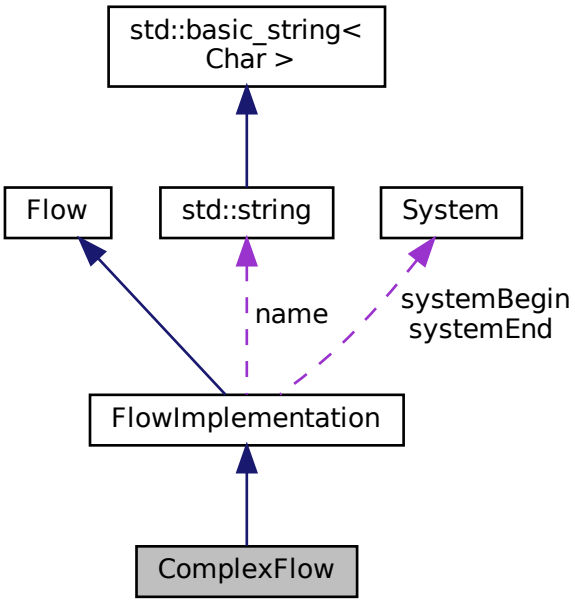
5.1 ComplexFlow Class Reference

`#include <FunctionalTests.h>`

Inheritance diagram for ComplexFlow:



Collaboration diagram for ComplexFlow:



Public Member Functions

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

Additional Inherited Members

5.1.1 Detailed Description

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

5.1.2 Constructor & Destructor Documentation

5.1.2.1 ComplexFlow()

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

Default constructor

Parameters

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

Returns

Complex flow with initial name, value, systemBegin and systemEnd

5.1.3 Member Function Documentation

5.1.3.1 expression()

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

Complex expression

Implements FlowImplementation.

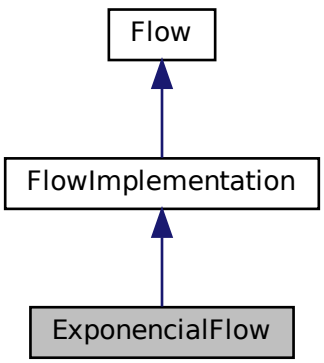
The documentation for this class was generated from the following file:

- test/functional/FunctionalTests.h

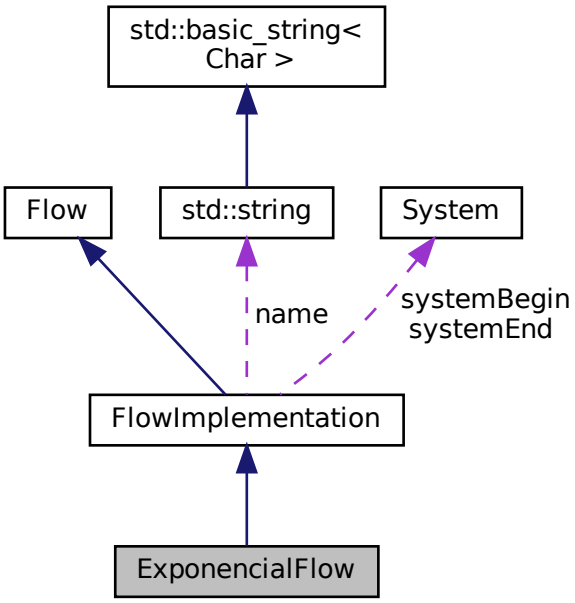
5.2 ExponencialFlow Class Reference

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

Inheritance diagram for ExponencialFlow:



Collaboration diagram for ExponencialFlow:



Public Member Functions

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

Additional Inherited Members

5.2.1 Detailed Description

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

5.2.2 Constructor & Destructor Documentation

5.2.2.1 ExponencialFlow()

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

Default constructor

Parameters

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

Returns

Exponencial flow with initial name, value, systemBegin and systemEnd

5.2.3 Member Function Documentation

5.2.3.1 expression()

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

Exponencial expression

Implements FlowImplementation.

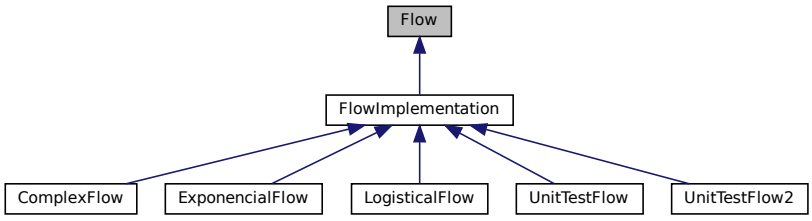
The documentation for this class was generated from the following file:

- test/functional/FunctionalTests.h

5.3 Flow Class Reference

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

Inheritance diagram for Flow:



Public Member Functions

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

5.3.1 Constructor & Destructor Documentation

5.3.1.1 ~Flow()

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

Default destructor

5.3.2 Member Function Documentation

5.3.2.1 expression()

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

Sets the expression of the flow

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

5.3.2.2 getName()

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

Get system name

Implemented in FlowImplementation.

5.3.2.3 getSystemBegin()

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

Get systemBegin

Implemented in FlowImplementation.

5.3.2.4 getSystemEnd()

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

Get systemEnd

Implemented in FlowImplementation.

5.3.2.5 getValue()

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

Get system value

Implemented in FlowImplementation.

5.3.2.6 setName()

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

Set system name

Parameters

n	Name for the flow

Implemented in FlowImplementation.

5.3.2.7 setSystemBegin()

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

Set systemBegin

Parameters

system	SystemBegin for the flow

Implemented in FlowImplementation.

5.3.2.8 setSystemEnd()

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

Set systemBegin

Parameters

system	SystemEnd for the flow

Implemented in FlowImplementation.

5.3.2.9 setValue()

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

Set system value

Parameters

v	Value for the flow

Implemented in FlowImplementation.

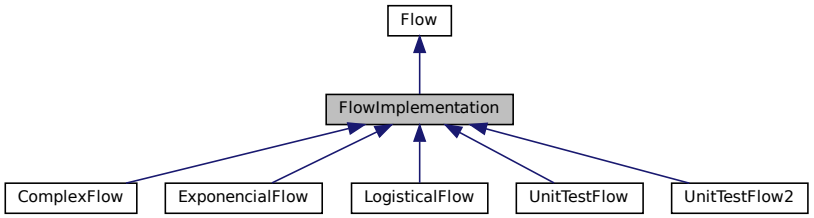
The documentation for this class was generated from the following file:

- src/lib/Flow.h

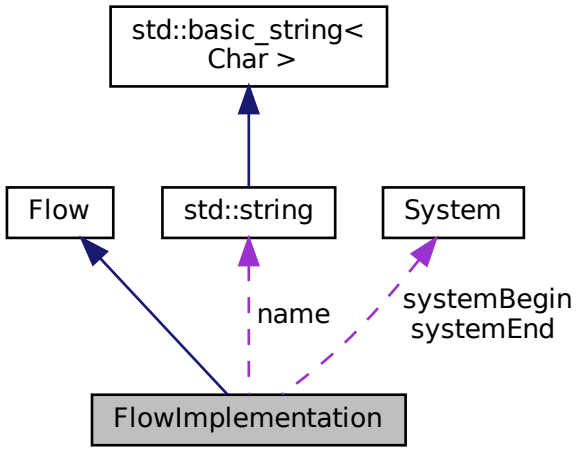
5.4 FlowImplementation Class Reference

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

Inheritance diagram for FlowImplementation:



Collaboration diagram for FlowImplementation:



Public Member Functions

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

Protected Attributes

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

5.4.1 Detailed Description

Flow that converges energy from a model to another

5.4.2 Constructor & Destructor Documentation

5.4.2.1 ~FlowImplementation()

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

Default destructor

5.4.2.2 FlowImplementation()

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

Default constructor

Parameters

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

Returns

Flow with initial name, value, systemBegin and systemEnd

5.4.3 Member Function Documentation

5.4.3.1 expression()

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

Sets the expression of the flow

Implements Flow.

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

5.4.3.2 getName()

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

Get system name

Implements Flow.

5.4.3.3 getSystemBegin()

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

Get systemBegin

Implements Flow.

5.4.3.4 getSystemEnd()

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

Get systemEnd

Implements Flow.

5.4.3.5 getValue()

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

Get system value

Implements Flow.

5.4.3.6 setName()

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

Set system name

Parameters

n	Name for the flow

Implements Flow.

5.4.3.7 setSystemBegin()

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

Set systemBegin

Parameters

system	SystemBegin for the flow

Implements Flow.

5.4.3.8 setSystemEnd()

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

Set systemBegin

Parameters

system	SystemEnd for the flow

Implements Flow.

5.4.3.9 setValue()

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

Set system value

Parameters

v	Value for the flow

Implements Flow.

5.4.4 Member Data Documentation

5.4.4.1 name

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

5.4.4.2 systemBegin

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

5.4.4.3 systemEnd

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

5.4.4.4 value

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

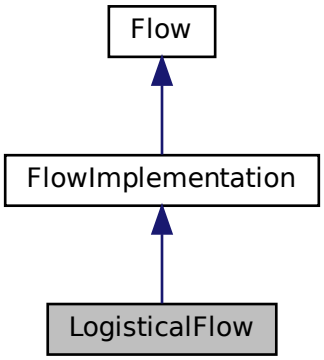
The documentation for this class was generated from the following files:

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

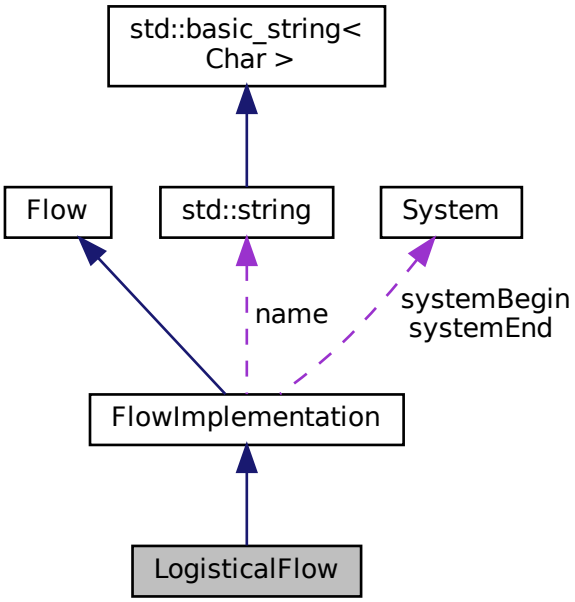
5.5 LogisticalFlow Class Reference

`#include <FunctionalTests.h>`

Inheritance diagram for LogisticalFlow:



Collaboration diagram for LogisticalFlow:



Public Member Functions

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

Additional Inherited Members

5.5.1 Detailed Description

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

5.5.2 Constructor & Destructor Documentation

5.5.2.1 LogisticalFlow()

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

Default constructor

Parameters

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

Returns

Logistical flow with initial name, value, systemBegin and systemEnd

5.5.3 Member Function Documentation

5.5.3.1 expression()

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

Logistical expression

Implements FlowImplementation.

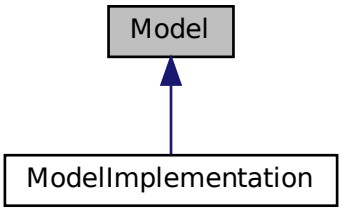
The documentation for this class was generated from the following file:

- test/functional/FunctionalTests.h

5.6 Model Class Reference

`#include <Model.h>`

Inheritance diagram for Model:



Public Member Functions

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

Static Public Member Functions

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

5.6.1 Constructor & Destructor Documentation

5.6.1.1 ~Model()

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

Default destructor

5.6.2 Member Function Documentation

5.6.2.1 add() [1/2]

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

Add a flow to the model

Parameters

flow	Flow to be added to the model

Implemented in ModelImplementation.

5.6.2.2 add() [2/2]

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

Add a system to the model

Parameters

system	System to be added to the model

Implemented in ModelImplementation.

5.6.2.3 createFlow()

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

5.6.2.4 createModel()

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

5.6.2.5 createSystem()

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

Implemented in ModelImplementation.

5.6.2.6 endFlows()

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

Implemented in ModelImplementation.

5.6.2.7 endModels()

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

Implemented in ModelImplementation.

5.6.2.8 endSystems()

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

Implemented in ModelImplementation.

5.6.2.9 getFlowsIterator()

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

Get model flows iterator

Implemented in ModelImplementation.

5.6.2.10 getModelsIterator()

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

Get model models iterator

Implemented in ModelImplementation.

5.6.2.11 getName()

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

Get model name

Implemented in ModelImplementation.

5.6.2.12 getSystemsIterator()

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

Get model systems iterator

Implemented in ModelImplementation.

5.6.2.13 getTime()

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

Get model time

Implemented in ModelImplementation.

5.6.2.14 setName()

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

Set model name

Parameters

n	Name for the system

Implemented in ModelImplementation.

5.6.2.15 setTime()

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

Set model time

Parameters

t	Name for the system

Implemented in ModelImplementation.

5.6.2.16 simulate()

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

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

Parameters

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

Implemented in ModelImplementation.

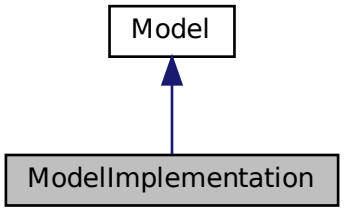
The documentation for this class was generated from the following files:

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

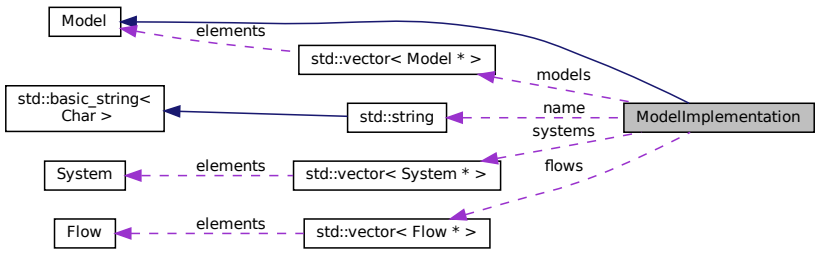
5.7 ModelImplementation Class Reference

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

Inheritance diagram for ModelImplementation:



Collaboration diagram for ModelImplementation:



Public Member Functions

- ~ModelImplementation () override
- ModelImplementation (std::string name, double time)
- void simulate (double start, double end, double timestep) override
- std::string getName () const override
- void setName (std::string n) override
- double getTime () const override
- void setTime (double t) override
- std::vector< System * >::iterator getSystemsIterator () override
- std::vector< Flow * >::iterator getFlowsIterator () override
- std::vector< Model * >::iterator getModelsIterator () override
- std::vector< System * >::iterator endSystems () override
- std::vector< Flow * >::iterator endFlows () override
- std::vector< Model * >::iterator endModels () override
- System * createSystem (std::string name, double value) override

Static Public Member Functions

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

Protected Member Functions

- void add (System *system) override
- void add (Flow *flow) override

Protected Attributes

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

Static Protected Attributes

- static std::vector< Model * > models

5.7.1 Detailed Description

Model that simulates the energy flow through models

5.7.2 Constructor & Destructor Documentation

5.7.2.1 ~ModelImplementation()

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

Default destructor destrutor padrao

5.7.2.2 ModelImplementation()

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

Default constructor

Parameters

name	Inital model name
time	Inital model time

Returns

Model with initial name and time

5.7.3 Member Function Documentation

5.7.3.1 add() [1/2]

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

Add a flow to the model

Parameters

flow	Flow to be added to the model

Implements Model.

5.7.3.2 add() [2/2]

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

Add a system to the model

Parameters

system	System to be added to the model

Implements Model.

5.7.3.3 createModel()

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

5.7.3.4 createSystem()

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

Implements Model.

5.7.3.5 endFlows()

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

Implements Model.

5.7.3.6 endModels()

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

Implements Model.

5.7.3.7 endSystems()

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

Implements Model.

5.7.3.8 getFlowsIterator()

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

Get model flows iterator

Implements Model.

5.7.3.9 getModelsIterator()

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

Get model models iterator

Implements Model.

5.7.3.10 getName()

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

Get model name

Implements Model.

5.7.3.11 getSystemsIterator()

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

Get model systems iterator

Implements Model.

5.7.3.12 getTime()

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

Get model time

Implements Model.

5.7.3.13 setName()

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

Set model name

Parameters

n	Name for the system

Implements Model.

5.7.3.14 setTime()

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

Set model time

Parameters

t	Name for the system

Implements Model.

5.7.3.15 simulate()

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

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

Parameters

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

Implements Model.

5.7.4 Member Data Documentation

5.7.4.1 flows

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

5.7.4.2 models

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

5.7.4.3 name

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

5.7.4.4 systems

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

5.7.4.5 time

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

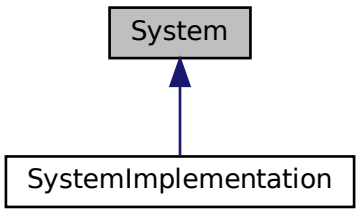
The documentation for this class was generated from the following files:

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

5.8 System Class Reference

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

Inheritance diagram for System:



Public Member Functions

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

5.8.1 Constructor & Destructor Documentation

5.8.1.1 ~System()

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

Default destructor

5.8.2 Member Function Documentation

5.8.2.1 getName()

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

Get system name

Implemented in SystemImplementation.

5.8.2.2 getValue()

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

Get system value

Implemented in SystemImplementation.

5.8.2.3 setName()

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

Set system name

Parameters

n	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.

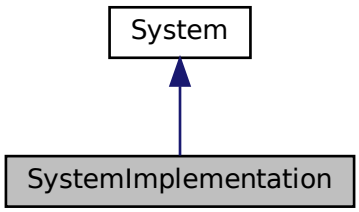
The documentation for this class was generated from the following file:

- src/lib/System.h

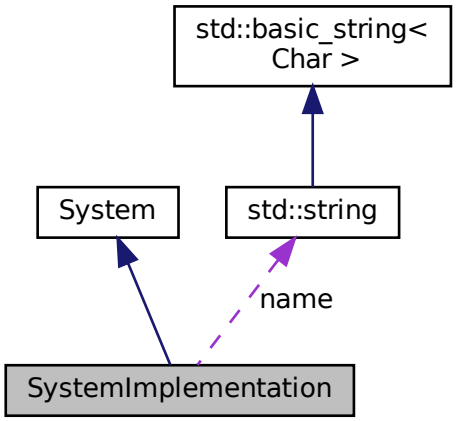
5.9 SystemImplementation Class Reference

`#include <SystemImplementation.h>`

Inheritance diagram for SystemImplementation:



Collaboration diagram for SystemImplementation:



Public Member Functions

- ~SystemImplementation () override
- SystemImplementation (std::string name, double value)
- 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

name	Inital system name
value	Inital system value

Returns

System with initial name and value

5.9.3 Member Function Documentation

5.9.3.1 getName()

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

Get system name

Implements System.

5.9.3.2 getValue()

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

Get system value

Implements System.

5.9.3.3 setName()

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

Set system name

Parameters

n	Name for the system

Implements System.

5.9.3.4 setValue()

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

Set system value

Parameters

v	Value for the system

Implements System.

5.9.4 Member Data Documentation

5.9.4.1 name

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

5.9.4.2 value

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

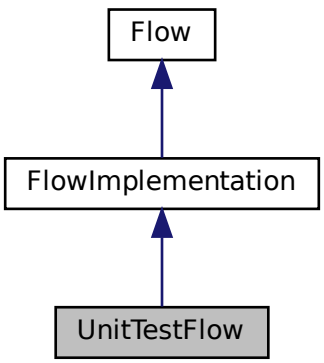
The documentation for this class was generated from the following files:

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

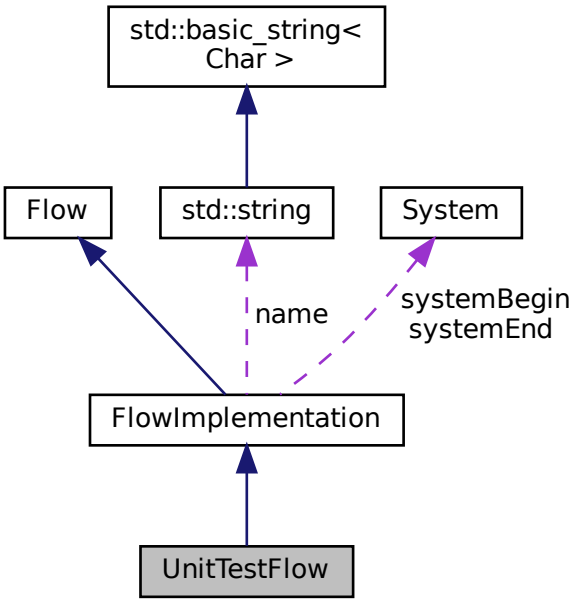
5.10 UnitTestFlow Class Reference

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

Inheritance diagram for UnitTestFlow:



Collaboration diagram for UnitTestFlow:



Public Member Functions

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

Additional Inherited Members

5.10.1 Detailed Description

Flow used for testing

5.10.2 Constructor & Destructor Documentation

5.10.2.1 UnitTestFlow()

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

Default constructor

Parameters

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

Returns

UnitTestFlow with initial name, value, systemBegin and systemEnd

5.10.3 Member Function Documentation

5.10.3.1 expression()

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

Flow expression method implementation for testing

Implements FlowImplementation.

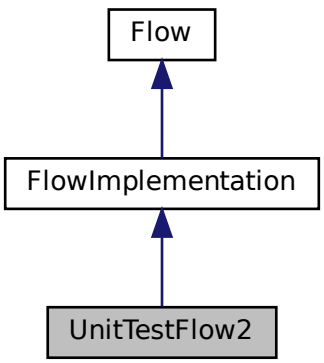
The documentation for this class was generated from the following file:

- test/unit/unitFlow.h

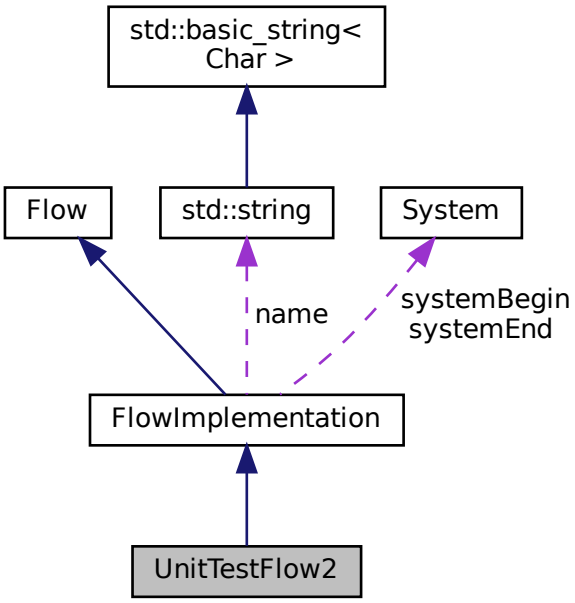
5.11 UnitTestFlow2 Class Reference

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

Inheritance diagram for UnitTestFlow2:



Collaboration diagram for UnitTestFlow2:



Public Member Functions

- UnitTestFlow2 (std::string name, System *systemBegin, System *systemEnd)
- double expression () override

Additional Inherited Members

5.11.1 Detailed Description

Flow used for testing

5.11.2 Constructor & Destructor Documentation

5.11.2.1 UnitTestFlow2()

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

Default constructor

Parameters

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

Returns

UnitTestFlow2 with initial name, value, systemBegin and systemEnd

5.11.3 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/CMake↩ CCompilerId.c File Reference

Macros

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

Functions

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

Variables

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

6.2.1 Macro Definition Documentation

6.2.1.1 __has_include

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

6.2.1.2 ARCHITECTURE_ID

```
#define ARCHITECTURE_ID
```

6.2.1.3 C_VERSION

```
#define C_VERSION
```

6.2.1.4 COMPILER_ID

```
#define COMPILER_ID ""
```

6.2.1.5 DEC

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

Value:

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

6.2.1.6 HEX

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

Value:

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

6.2.1.7 PLATFORM_ID

```
#define PLATFORM_ID
```

6.2.1.8 STRINGIFY

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

6.2.1.9 STRINGIFY_HELPER

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

6.2.2 Function Documentation

6.2.2.1 main()

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

6.2.3 Variable Documentation

6.2.3.1 info_arch

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

6.2.3.2 info_compiler

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

6.2.3.3 info_language_extensions_default

```
const char* info_language_extensions_default
```

Initial value:

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

6.2.3.4 info_language_standard_default

```
const char* info_language_standard_default
```

Initial value:

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

6.2.3.5 info_platform

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

6.3 cmake-build-debug/CMakeFiles/3.22.3/CompilerIdCXX/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

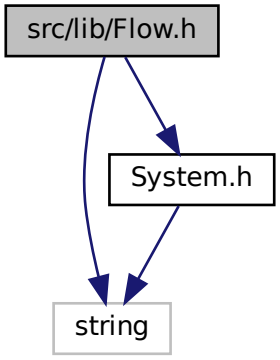
6.7 CMakeLists.txt File Reference

6.8 README.md File Reference

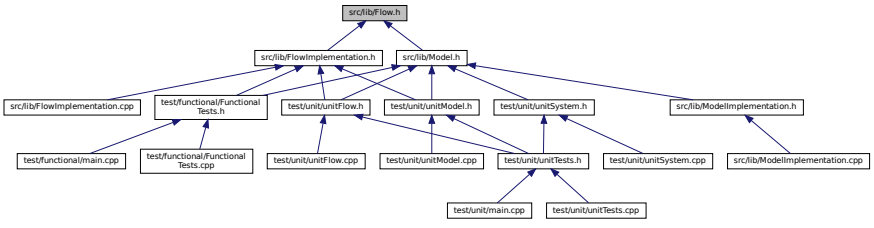
6.9 src/lib/Flow.h File Reference

```
#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

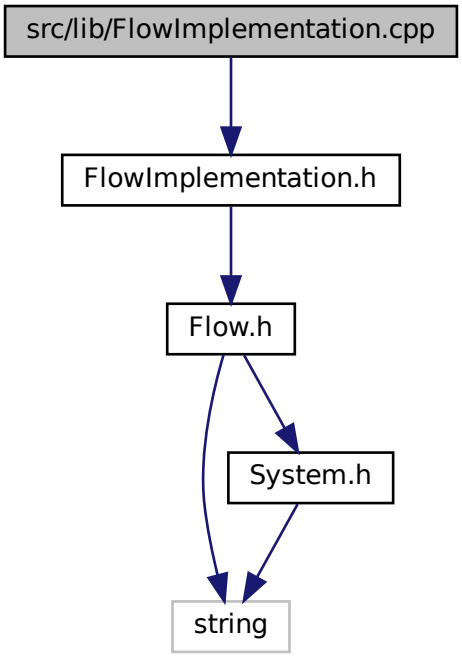
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 27/04/2022.
3 //
4
5 #ifndef ENG1_FLOW_H
6 #define ENG1_FLOW_H
7
8 #include <string>
9
10 #include "System.h"
11
12 class Flow {
13 public:
17     virtual ~Flow() = default;
21     virtual std::string getName() const = 0;
26     virtual void setName(std::string) = 0;
30     virtual double getValue() const = 0;
35     virtual void setValue(double) = 0;
39     virtual double expression() = 0;
43     virtual System* getSystemBegin() const = 0;
48     virtual void setSystemBegin(System*) = 0;
52     virtual System* getSystemEnd() const = 0;
57     virtual void setSystemEnd(System*) = 0;
58 };
59
60
61 #endif //ENG1_FLOW_H
```

6.11 src/lib/FlowImplementation.cpp File Reference

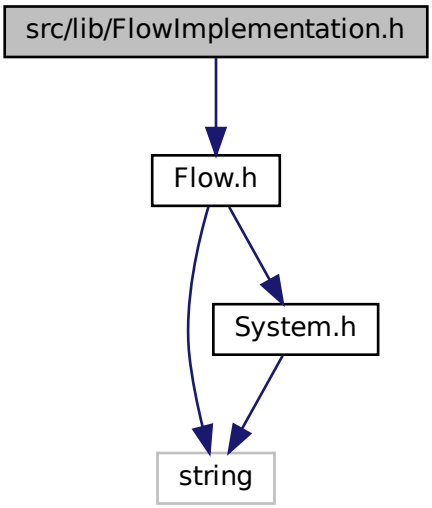
```
#include "FlowImplementation.h"
```

Include dependency graph for FlowImplementation.cpp:

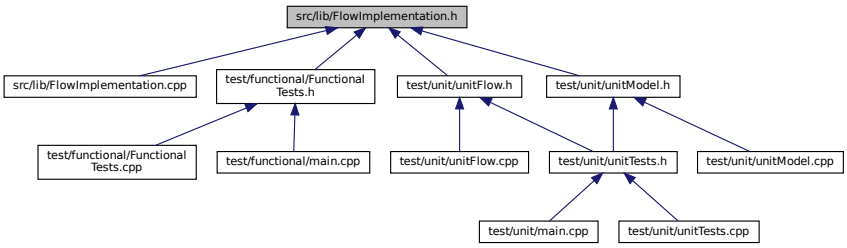


6.12 src/lib/FlowImplementation.h File Reference

`#include "Flow.h"`
Include 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.

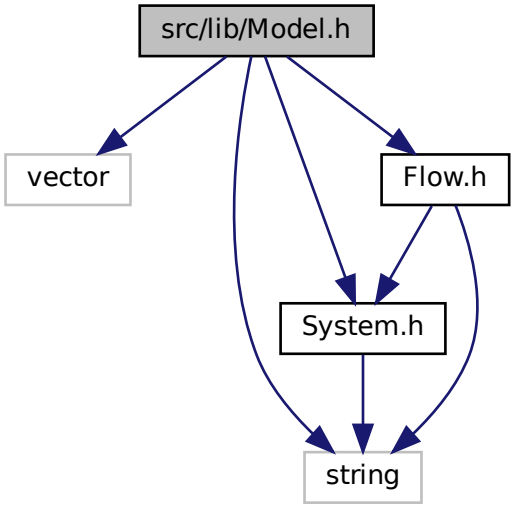
```
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
```

```
4
5 #ifndef ENG1_FLOWIMPLEMENTATION_H
6 #define ENG1_FLOWIMPLEMENTATION_H
7
8
9 #include "Flow.h"
10
14 class FlowImplementation : public Flow {
15
16 private:
22     FlowImplementation(const FlowImplementation& flow);
23
29     FlowImplementation& operator=(const FlowImplementation& flow);
30
31 protected:
32     std::string name;
33     double value;
34     System* systemBegin;
35     System* systemEnd;
36
37 public:
41     ~FlowImplementation() override;
42
51     FlowImplementation(std::string name, System* systemBegin, System* systemEnd);
52
56     double expression() override = 0;
57
61     std::string getName() const override;
62
67     void setName(std::string n) override;
68
72     double getValue() const override;
73
78     void setValue(double v) override;
79
83     System* getSystemBegin() const override;
84
89     void setSystemBegin(System* system) override;
90
94     System* getSystemEnd() const override;
95
100     void setSystemEnd(System* system) override;
101 };
102
103
104 #endif //ENG1_FLOWIMPLEMENTATION_H
```

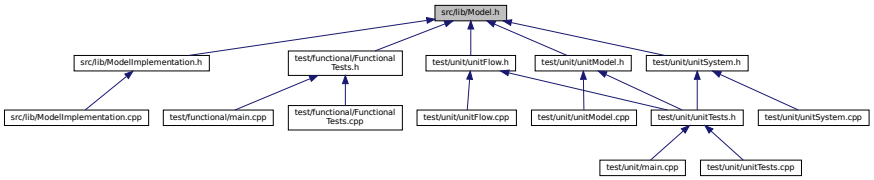
6.14 src/lib/Model.h File Reference

```
#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.

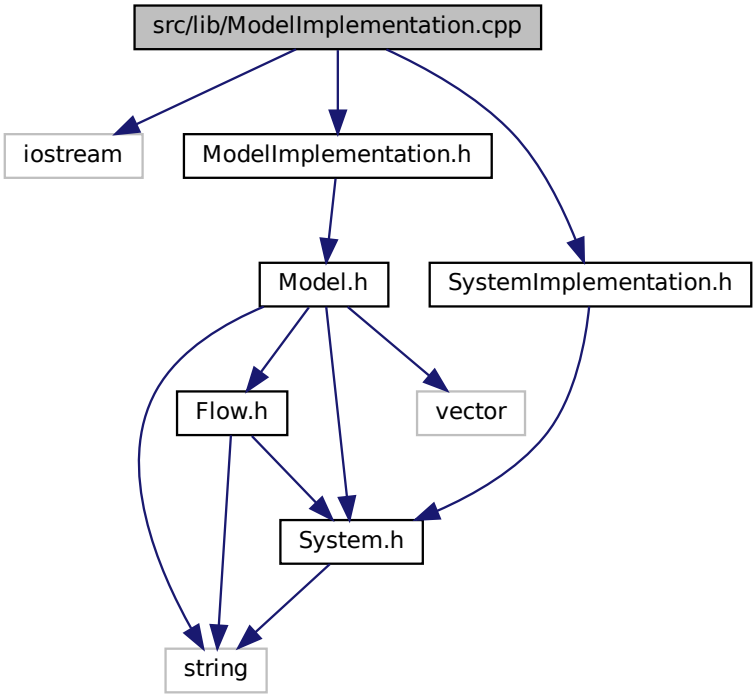
```
1 //
2 // Created by joaozenobio on 27/04/2022.
3 //
4
5 #ifndef ENG1_MODEL_H
6 #define ENG1_MODEL_H
7
8
9 #include <vector>
10 #include <string>
11
12 #include "System.h"
13 #include "Flow.h"
```

```
14
15 class Model {
16 public:
20     virtual ~Model() = default;
27     virtual void simulate(double, double, double) = 0;
31     virtual std::string getName() const = 0;
36     virtual void setName(std::string) = 0;
40     virtual double getTime() const = 0;
45     virtual void setTime(double) = 0;
50     virtual void add(System*) = 0;
55     virtual void add(Flow*) = 0;
59     virtual std::vector<System*>::iterator getSystemsIterator() = 0;
63     virtual std::vector<Flow*>::iterator getFlowsIterator() = 0;
67     virtual std::vector<Model*>::iterator getModelsIterator() = 0;
68
69     virtual std::vector<System*>::iterator endSystems() = 0;
70
71     virtual std::vector<Flow*>::iterator endFlows() = 0;
72
73     virtual std::vector<Model*>::iterator endModels() = 0;
74
75     virtual System* createSystem(std::string name, double value) = 0;
76
77     template<typename FlowType>
78     Flow* createFlow(std::string name, System* systemBegin, System* systemEnd){
79         Flow* flow = new FlowType(name, systemBegin, systemEnd);
80         add(flow);
81         return flow;
82     }
83
84     static Model* createModel(std::string name, double time);
85 };
86
87
88 #endif //ENG1_MODEL_H
```

6.16 src/lib/ModelImplementation.cpp File Reference

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

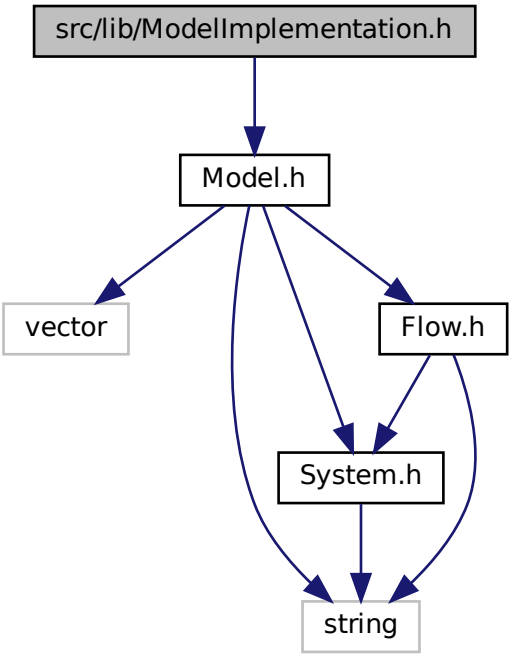
Include dependency graph for ModelImplementation.cpp:



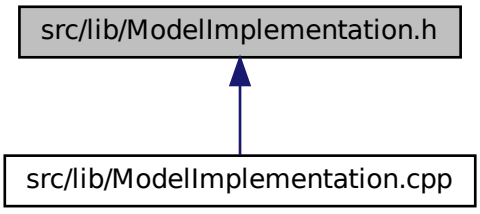
6.17 src/lib/ModelImplementation.h File Reference

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

Include dependency graph for ModelImplementation.h:



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



Classes

- class ModelImplementation

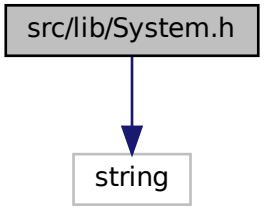
6.18 ModelImplementation.h

Go to the documentation of this file.

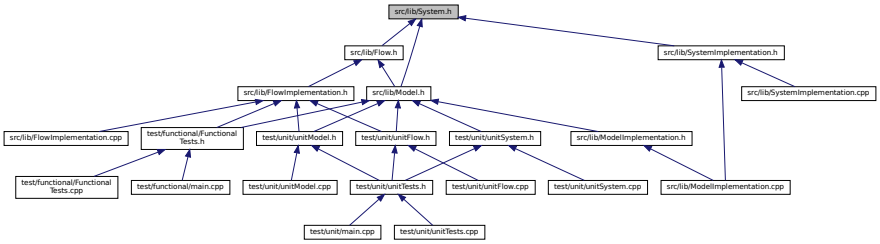
```
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENG1_MODELIMPLEMENTATION_H
6 #define ENG1_MODELIMPLEMENTATION_H
7
8
9 #include "Model.h"
10
14 class ModelImplementation: public Model {
15
16 private:
22     ModelImplementation(const ModelImplementation& model);
23
29     ModelImplementation& operator=(const ModelImplementation& model);
30
31 protected:
32     std::string name;
33     double time;
34     std::vector<System*> systems;
35     std::vector<Flow*> flows;
36     static std::vector<Model*> models;
41     void add(System* system) override;
42
47     void add(Flow* flow) override;
48
49 public:
53     ~ModelImplementation() override;
54
61     ModelImplementation(std::string name, double time);
62
69     void simulate(double start, double end, double timestep) override;
70
74     std::string getName() const override;
75
80     void setName(std::string n) override;
81
85     double getTime() const override;
86
91     void setTime(double t) override;
92
96     std::vector<System*>::iterator getSystemsIterator() override;
97
101     std::vector<Flow*>::iterator getFlowsIterator() override;
102
106     std::vector<Model*>::iterator getModelsIterator() override;
107
108     std::vector<System*>::iterator endSystems() override;
109
110     std::vector<Flow*>::iterator endFlows() override;
111
112     std::vector<Model*>::iterator endModels() override;
113
114     System* createSystem(std::string name, double value) override;
115
116     static Model* createModel(std::string name, double time);
117 };
118
119
120 #endif //ENG1_MODELIMPLEMENTATION_H
```

6.19 src/lib/System.h File Reference

`#include <string>`
Include dependency graph for System.h:



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



Classes

- class System

6.20 System.h

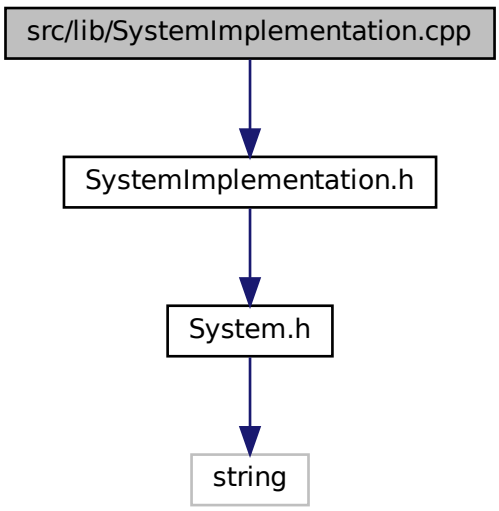
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 27/04/2022.
3 //
4
5 #ifndef ENG1_SYSTEM_H
6 #define ENG1_SYSTEM_H
7
8 #include <string>
9
10 class System {
11 public:
15     virtual ~System() = default;
19     virtual std::string getName() const = 0;
24     virtual void setName(std::string) = 0;
28     virtual double getValue() const = 0;
33     virtual void setValue(double) = 0;
34 };
35
36
37 #endif //ENG1_SYSTEM_H
```

6.21 src/lib/SystemImplementation.cpp File Reference

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

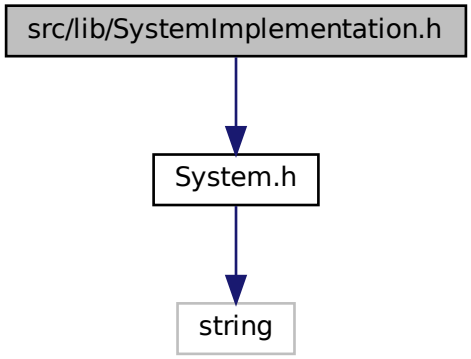
Include dependency graph for SystemImplementation.cpp:



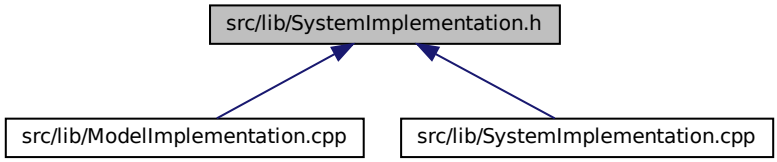
6.22 src/lib/SystemImplementation.h File Reference

```
#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

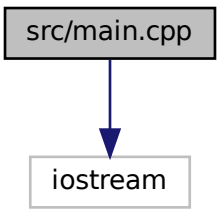
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENG1_SYSTEMIMPLEMENTATION_H
6 #define ENG1_SYSTEMIMPLEMENTATION_H
7
8 #include "System.h"
9
13 class SystemImplementation : public System {
14
15 private:
21     SystemImplementation(const SystemImplementation& system);
22
28     SystemImplementation& operator=(const SystemImplementation& system);
29
30 protected:
31     std::string name;
32     double value;
33
34 public:
38     ~SystemImplementation() override;
39
46     SystemImplementation(std::string name, double value);
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 //ENG1_SYSTEMIMPLEMENTATION_H
```

6.24 src/main.cpp File Reference

```
#include <iostream>
```

Include dependency graph for main.cpp:



Functions

- int main ()

6.24.1 Function Documentation

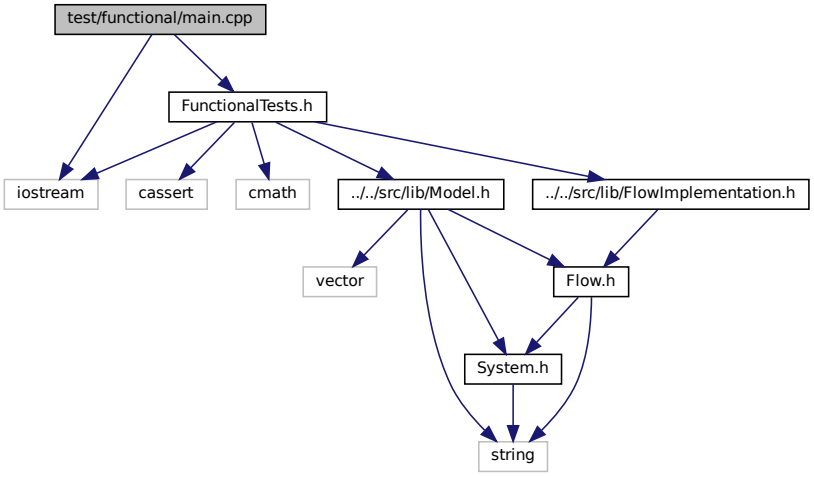
6.24.1.1 main()

```
int main ( )
```

6.25 test/functional/main.cpp File Reference

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

Include dependency graph for main.cpp:



Functions

- int main ()

6.25.1 Function Documentation

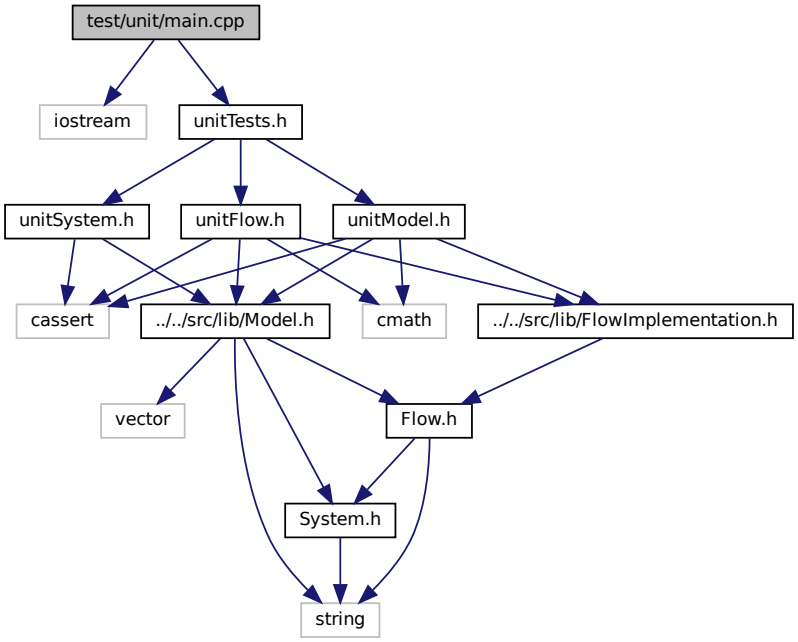
6.25.1.1 main()

```
int main ( )
```

6.26 test/unit/main.cpp File Reference

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

Include dependency graph for main.cpp:



Functions

- int main ()

6.26.1 Function Documentation

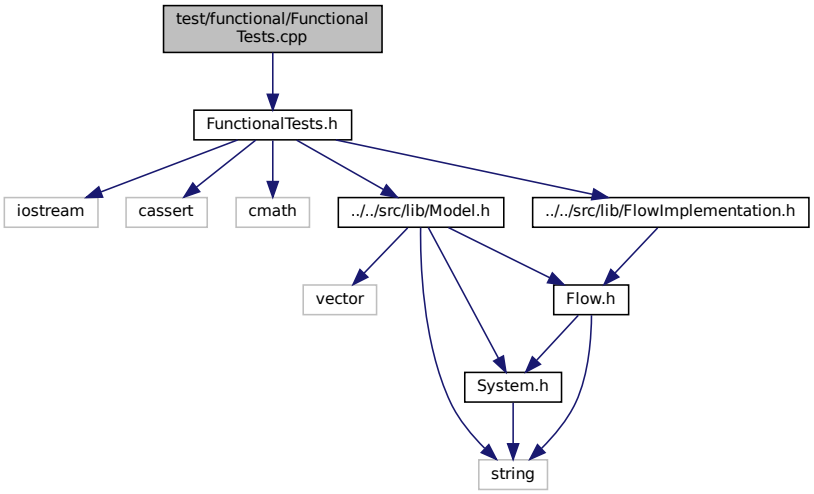
6.26.1.1 main()

```
int main ( )
```

6.27 test/functional/FunctionalTests.cpp File Reference

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

Include dependency graph for FunctionalTests.cpp:



Functions

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

6.27.1 Function Documentation

6.27.1.1 ComplexTest()

```
void ComplexTest ( )
```

Function to test the complex flow.

6.27.1.2 ExponencialTest()

```
void ExponencialTest ( )
```

Function to test the exponencial flow.

6.27.1.3 LogisticalTest()

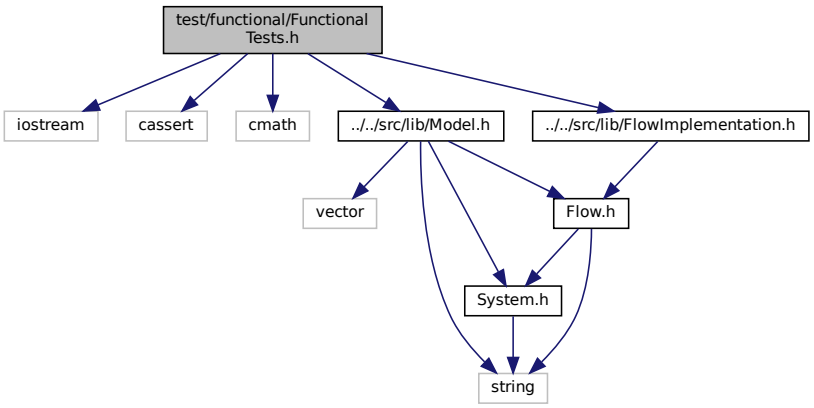
```
void LogisticalTest ( )
```

Function to test the logistical flow.

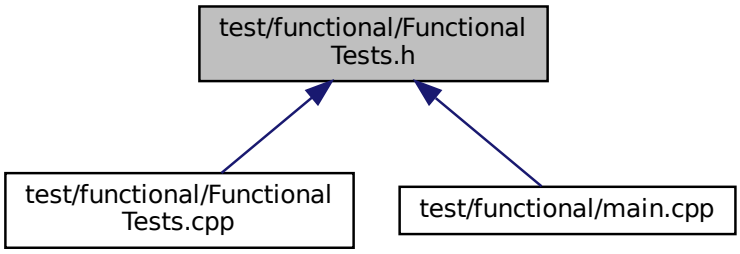
6.28 test/functional/FunctionalTests.h File Reference

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

Include dependency graph for FunctionalTests.h:



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



Classes

- class ExponencialFlow
- class LogisticalFlow
- class ComplexFlow

Functions

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

6.28.1 Function Documentation

6.28.1.1 ComplexTest()

```
void ComplexTest ( )
```

Function to test the complex flow.

6.28.1.2 ExponencialTest()

```
void ExponencialTest ( )
```

Function to test the exponencial flow.

6.28.1.3 LogisticalTest()

```
void LogisticalTest ( )
```

Function to test the logistical flow.

6.29 FunctionalTests.h

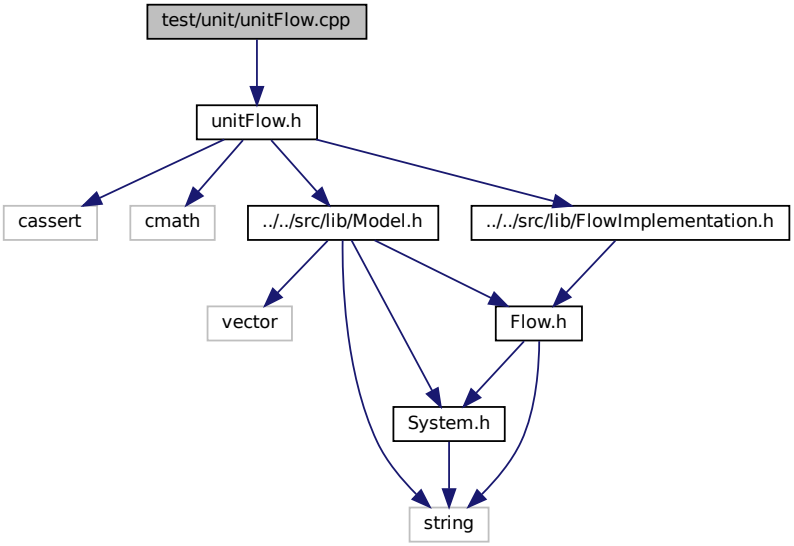
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 28/04/2022.
3 //
4
5 #ifndef ENG1_FUNCTIONALTESTS_H
6 #define ENG1_FUNCTIONALTESTS_H
7
8 #include <iostream>
9 #include <cassert>
10 #include <cmath>
11
12 #include "../../src/lib/Model.h"
13 #include "../../src/lib/FlowImplementation.h"
14
18 class ExponencialFlow : public FlowImplementation{
19 public:
28     ExponencialFlow(std::string name, System* systemOut, System* systemIn) : FlowImplementation(name,
      systemOut, systemIn) {}
32     double expression() override {
33         return 0.01 * getSystemBegin()->getValue();
34     }
35 };
36
41 class LogisticalFlow : public FlowImplementation{
42 public:
```

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

- void unitFlowSetSystemBegin ()
- void unitFlowGetSystemEnd ()
- void unitFlowSetSystemEnd ()
- void runUnitTestsFlow ()

6.30.1 Function Documentation

6.30.1.1 runUnitTestsFlow()

```
void runUnitTestsFlow ( )
```

Run all unit tests for Flow

6.30.1.2 unitFlowDefaultConstructor()

```
void unitFlowDefaultConstructor ( )
```

Tests Flow default constructor

6.30.1.3 unitFlowDestructor()

```
void unitFlowDestructor ( )
```

Tests Flow destructor

6.30.1.4 unitFlowExpression()

```
void unitFlowExpression ( )
```

Tests Flow expression

6.30.1.5 unitFlowGetName()

```
void unitFlowGetName ( )
```

Tests Flow getName method

6.30.1.6 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests Flow getSystemBegin method

6.30.1.7 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests Flow getSystemEnd method

6.30.1.8 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests Flow getValue method

6.30.1.9 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests Flow setName method

6.30.1.10 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests Flow setSystemBegin method

6.30.1.11 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests Flow setSystemEnd method

6.30.1.12 unitFlowSetValue()

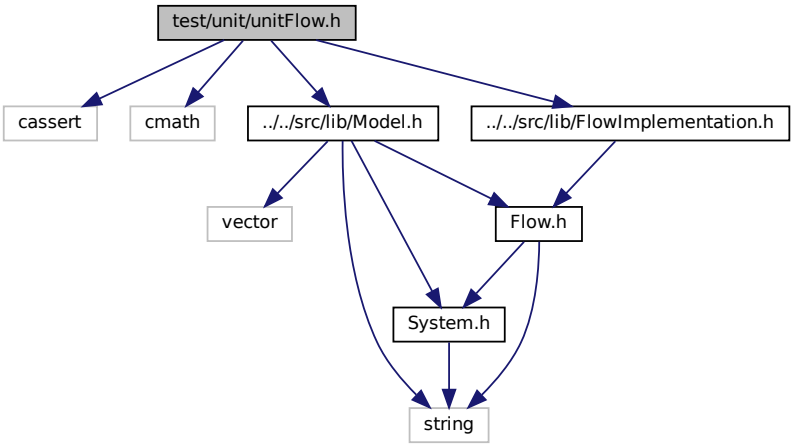
```
void unitFlowSetValue ( )
```

Tests Flow setValue method

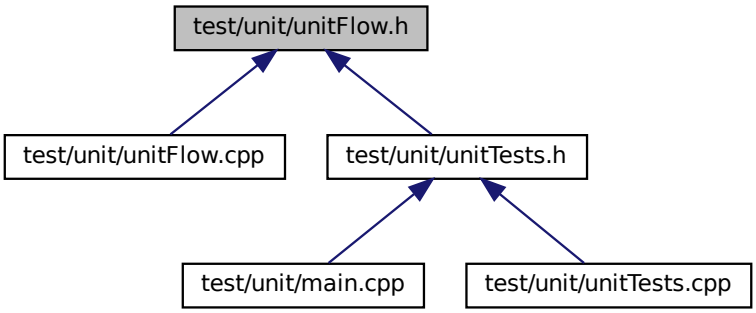
6.31 test/unit/unitFlow.h File Reference

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

Include dependency graph for unitFlow.h:



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



Classes

- class UnitTestFlow

Functions

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

6.31.1 Function Documentation

6.31.1.1 runUnitTestsFlow()

```
void runUnitTestsFlow ( )
```

Run all unit tests for Flow

6.31.1.2 unitFlowDefaultConstructor()

```
void unitFlowDefaultConstructor ( )
```

Tests Flow default constructor

6.31.1.3 unitFlowDestructor()

```
void unitFlowDestructor ( )
```

Tests Flow destructor

6.31.1.4 unitFlowExpression()

```
void unitFlowExpression ( )
```

Tests Flow expression

6.31.1.5 unitFlowGetName()

```
void unitFlowGetName ( )
```

Tests Flow getName method

6.31.1.6 unitFlowGetSystemBegin()

```
void unitFlowGetSystemBegin ( )
```

Tests Flow getSystemBegin method

6.31.1.7 unitFlowGetSystemEnd()

```
void unitFlowGetSystemEnd ( )
```

Tests Flow getSystemEnd method

6.31.1.8 unitFlowGetValue()

```
void unitFlowGetValue ( )
```

Tests Flow getValue method

6.31.1.9 unitFlowSetName()

```
void unitFlowSetName ( )
```

Tests Flow setName method

6.31.1.10 unitFlowSetSystemBegin()

```
void unitFlowSetSystemBegin ( )
```

Tests Flow setSystemBegin method

6.31.1.11 unitFlowSetSystemEnd()

```
void unitFlowSetSystemEnd ( )
```

Tests Flow setSystemEnd method

6.31.1.12 unitFlowSetValue()

```
void unitFlowSetValue ( )
```

Tests Flow setValue method

6.32 unitFlow.h

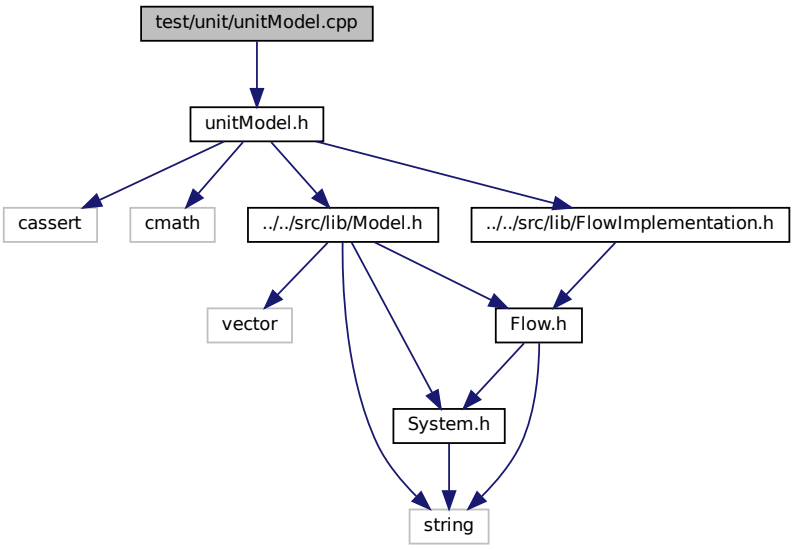
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENG1_UNITFLOW_H
6 #define ENG1_UNITFLOW_H
7
8
9 #include <cassert>
10 #include <cmath>
11
12 #include "../../src/lib/Model.h"
13 #include "../../src/lib/FlowImplementation.h"
14
18 class UnitTestFlow : public FlowImplementation{
19 public:
28     UnitTestFlow(std::string name, System* systemBegin, System* systemEnd) : FlowImplementation(name,
      systemBegin, systemEnd) {}
32     double expression() override {
33         return 0.01 * getSystemBegin()->getValue();
34     }
35 };
36
37 void unitFlowDestructor();
38 void unitFlowDefaultConstructor();
39 void unitFlowExpression();
40 void unitFlowGetName();
41 void unitFlowSetName();
42 void unitFlowGetValue();
43 void unitFlowSetValue();
44 void unitFlowGetSystemBegin();
45 void unitFlowSetSystemBegin();
46 void unitFlowGetSystemEnd();
47 void unitFlowSetSystemEnd();
48 void runUnitTestsFlow();
49
50
51 #endif //ENG1_UNITFLOW_H
```

6.33 test/unit/unitModel.cpp File Reference

`#include "unitModel.h"`

Include dependency graph for unitModel.cpp:



Functions

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

6.33.1 Function Documentation

6.33.1.1 runUnitTestsModel()

```
void runUnitTestsModel ( )
```

Run all unit tests for Model

6.33.1.2 unitModelAddFlow()

```
void unitModelAddFlow ( )
```

Tests Model add to add a Flow

6.33.1.3 unitModelAddSystem()

```
void unitModelAddSystem ( )
```

Tests Model add to add a System

6.33.1.4 unitModelCreateFlow()

```
void unitModelCreateFlow ( )
```

Tests Model createFlow

6.33.1.5 unitModelCreateModel()

```
void unitModelCreateModel ( )
```

Tests Model createModel

6.33.1.6 unitModelCreateSystem()

```
void unitModelCreateSystem ( )
```

Tests Model createSystem

6.33.1.7 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests Model default constructor

6.33.1.8 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests Model destructor

6.33.1.9 unitModelGetName()

```
void unitModelGetName ( )
```

Tests Model getName method

6.33.1.10 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests Model getTime method

6.33.1.11 unitModelSetName()

```
void unitModelSetName ( )
```

Tests Model setName method

6.33.1.12 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests Model setTime method

6.33.1.13 unitModelSimulate()

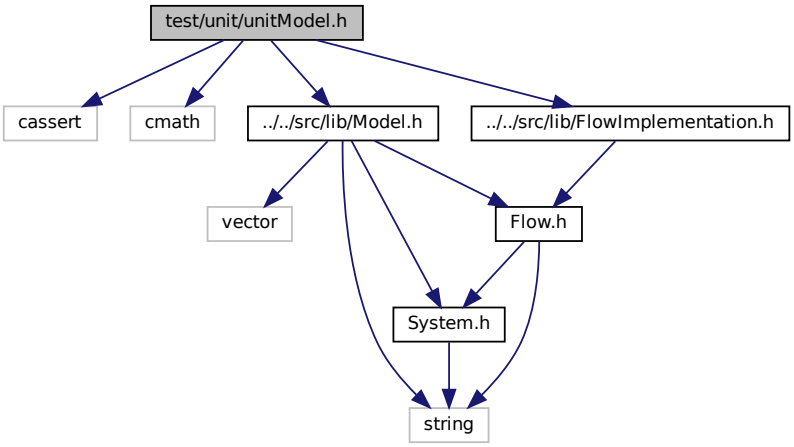
```
void unitModelSimulate ( )
```

Tests Model simluate method

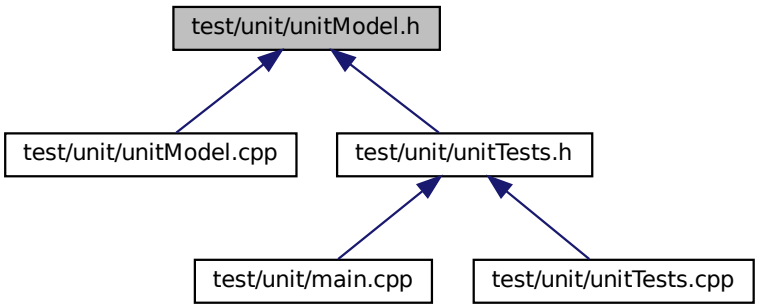
6.34 test/unit/unitModel.h File Reference

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

Include dependency graph for unitModel.h:



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



Classes

- class UnitTestFlow2

Functions

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

6.34.1 Function Documentation

6.34.1.1 runUnitTestsModel()

```
void runUnitTestsModel ( )
```

Run all unit tests for Model

6.34.1.2 unitModelAddFlow()

```
void unitModelAddFlow ( )
```

Tests Model add to add a Flow

6.34.1.3 unitModelAddSystem()

```
void unitModelAddSystem ( )
```

Tests Model add to add a System

6.34.1.4 unitModelCreateFlow()

```
void unitModelCreateFlow ( )
```

Tests Model createFlow

6.34.1.5 unitModelCreateModel()

```
void unitModelCreateModel ( )
```

Tests Model createModel

6.34.1.6 unitModelCreateSystem()

```
void unitModelCreateSystem ( )
```

Tests Model createSystem

6.34.1.7 unitModelDefaultConstructor()

```
void unitModelDefaultConstructor ( )
```

Tests Model default constructor

6.34.1.8 unitModelDestructor()

```
void unitModelDestructor ( )
```

Tests Model destructor

6.34.1.9 unitModelGetName()

```
void unitModelGetName ( )
```

Tests Model getName method

6.34.1.10 unitModelGetTime()

```
void unitModelGetTime ( )
```

Tests Model getTime method

6.34.1.11 unitModelSetName()

```
void unitModelSetName ( )
```

Tests Model setName method

6.34.1.12 unitModelSetTime()

```
void unitModelSetTime ( )
```

Tests Model setTime method

6.34.1.13 unitModelSimulate()

```
void unitModelSimulate ( )
```

Tests Model simluate method

6.35 unitModel.h

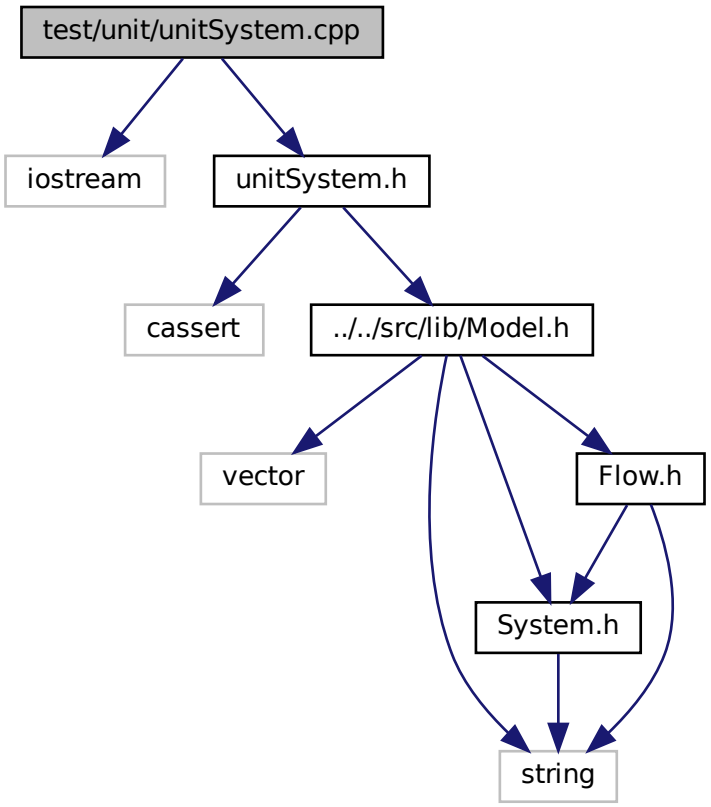
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENG1_UNITMODEL_H
6 #define ENG1_UNITMODEL_H
7
8
9 #include <cassert>
10 #include <cmath>
11
12 #include "../../src/lib/Model.h"
13 #include "../../src/lib/FlowImplementation.h"
14
18 class UnitTestFlow2 : public FlowImplementation{
19 public:
28     UnitTestFlow2(std::string name, System* systemBegin, System* systemEnd) : FlowImplementation(name,
      systemBegin, systemEnd) {}
32     double expression() override {
33         return 0.01 * getSystemBegin()->getValue();
34     }
35 };
36
37 void unitModelDestructor();
38 void unitModelDefaultConstructor();
39 void unitModelSimulate();
40 void unitModelGetName();
41 void unitModelSetName();
42 void unitModelGetTime();
43 void unitModelSetTime();
44 void unitModelAddSystem();
45 void unitModelAddFlow();
46 void unitModelCreateSystem();
47 void unitModelCreateFlow();
48 void unitModelCreateModel();
49 void runUnitTestsModel();
50
51
52 #endif //ENG1_UNITMODEL_H
```

6.36 test/unit/unitSystem.cpp File Reference

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

Include dependency graph for unitSystem.cpp:



Functions

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

6.36.1 Function Documentation

6.36.1.1 runUnitTestsSystem()

```
void runUnitTestsSystem ( )
```

Run all unit tests for System

6.36.1.2 unitSystemDefaultConstructor()

```
void unitSystemDefaultConstructor ( )
```

Tests System default constructor

6.36.1.3 unitSystemDestructor()

```
void unitSystemDestructor ( )
```

Tests System destructor

6.36.1.4 unitSystemGetName()

```
void unitSystemGetName ( )
```

Tests System getName method

6.36.1.5 unitSystemGetValue()

```
void unitSystemGetValue ( )
```

Tests System getValue method

6.36.1.6 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests System setName method

6.36.1.7 unitSystemSetValue()

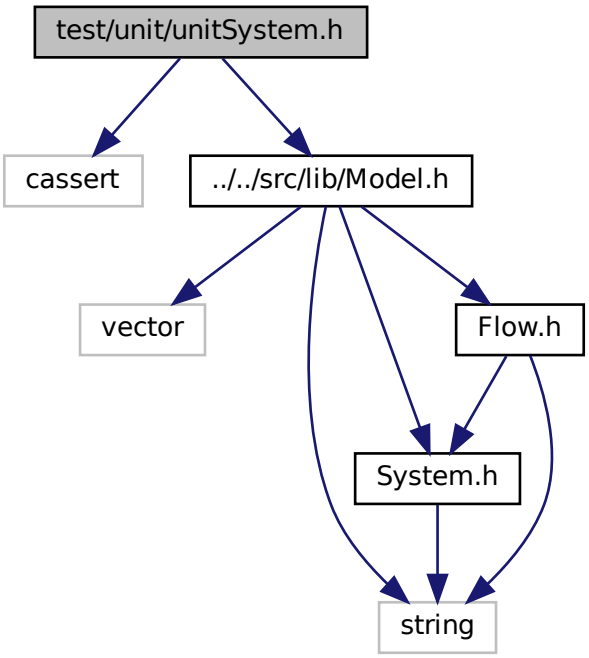
```
void unitSystemSetValue ( )
```

Tests System setValue method

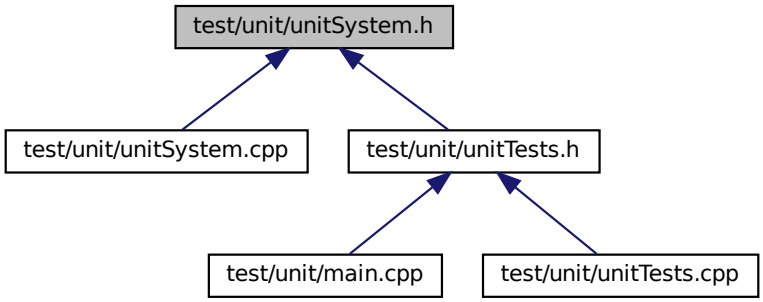
6.37 test/unit/unitSystem.h File Reference

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

Include dependency graph for unitSystem.h:



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



Functions

- void unitSystemDestructor ()

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

6.37.1 Function Documentation

6.37.1.1 runUnitTestsSystem()

```
void runUnitTestsSystem ( )
```

Run all unit tests for System

6.37.1.2 unitSystemDefaultConstructor()

```
void unitSystemDefaultConstructor ( )
```

Tests System default constructor

6.37.1.3 unitSystemDestructor()

```
void unitSystemDestructor ( )
```

Tests System destructor

6.37.1.4 unitSystemGetName()

```
void unitSystemGetName ( )
```

Tests System getName method

6.37.1.5 unitSystemGetValue()

```
void unitSystemGetValue ( )
```

Tests System getValue method

6.37.1.6 unitSystemSetName()

```
void unitSystemSetName ( )
```

Tests System setName method

6.37.1.7 unitSystemSetValue()

```
void unitSystemSetValue ( )
```

Tests System setValue method

6.38 unitSystem.h

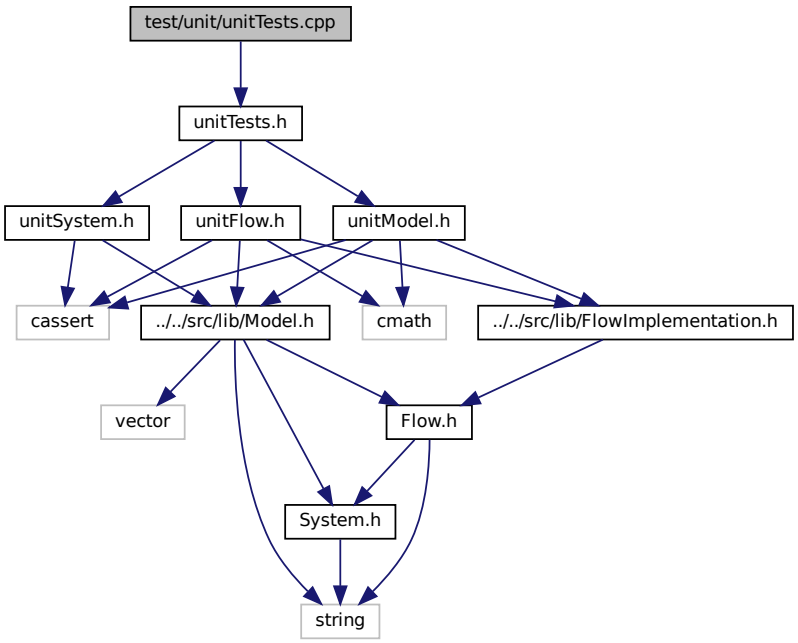
Go to the documentation of this file.

```
1 //
2 // Created by joaozenobio on 19/05/22.
3 //
4
5 #ifndef ENG1_UNITSYSTEM_H
6 #define ENG1_UNITSYSTEM_H
7
8 #include <cassert>
9
10 #include "../../src/lib/Model.h"
11
12 void unitSystemDestructor();
13 void unitSystemDefaultConstructor();
14 void unitSystemGetName();
15 void unitSystemSetName();
16 void unitSystemGetValue();
17 void unitSystemSetValue();
18 void runUnitTestsSystem();
19
20 #endif //ENG1_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

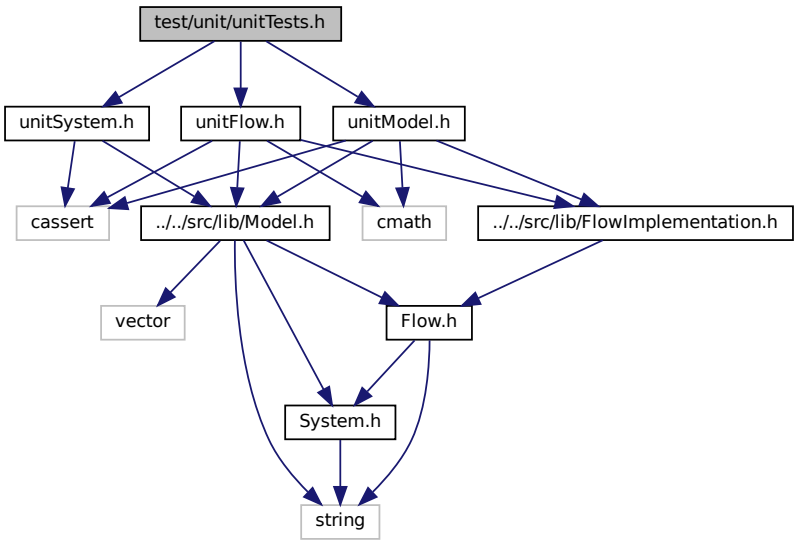
Tests Flow methods

Tests Model methods

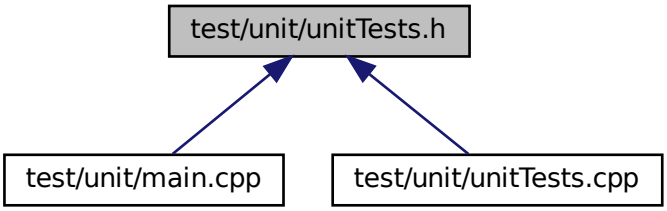
6.40 test/unit/unitTests.h File Reference

```
#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 ENG1_UNITTESTS_H
6 #define ENG1_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 //ENG1_UNITTESTS_H
```

Index