# FutureSynchronyzer /doc

Generated by Doxygen 1.8.9.1

Wed Feb 25 2015 21:48:59

# **Contents**

1	Nam	nespace	Index	1
	1.1	Names	space List	1
2	Hier	archical	I Index	3
	2.1	Class I	Hierarchy	3
3	Clas	s Index		5
	3.1	Class L	List	5
4	File	Index		7
	4.1	File Lis	st	7
5	Nam	nespace	Documentation	9
	5.1	factory	Namespace Reference	9
		5.1.1	Detailed Description	9
	5.2	policy I	Namespace Reference	9
		5.2.1	Detailed Description	10
		5.2.2	Typedef Documentation	10
			5.2.2.1 NoResultPolicy	10
	5.3	thread	Namespace Reference	10
		5.3.1	Detailed Description	10
6	Clas	s Docui	mentation	11
	6.1	policy::	:AddNumberResultPolicy Class Reference	11
		6.1.1	Detailed Description	11
		6.1.2	Constructor & Destructor Documentation	11
			6.1.2.1 AddNumberResultPolicy	11
			6.1.2.2 ~AddNumberResultPolicy	11
		6.1.3	Member Function Documentation	12
			6.1.3.1 applyResult	12
	6.2	factory	:::AsyncFutureFactory Class Reference	12
		6.2.1	Detailed Description	12
		600	Constructor & Destructor Desumentation	10

iv CONTENTS

		6.2.2.1	AsyncFutureFactory	12
		6.2.2.2	~AsyncFutureFactory	12
	6.2.3	Member	Function Documentation	12
		6.2.3.1	createFuture	12
6.3	Futurel	Factory Cla	ass Reference	13
6.4	thread:	:FutureSy	${\tt nchronyzer} < {\tt ResultPolicy}, \ {\tt FutureFactory}, \ {\tt OutputResult} > {\tt Class\ Template\ Reference}$	14
	6.4.1	Detailed	Description	16
	6.4.2	Construc	tor & Destructor Documentation	16
		6.4.2.1	FutureSynchronyzer	16
		6.4.2.2	$\sim$ FutureSynchronyzer	17
	6.4.3	Member	Function Documentation	17
		6.4.3.1	addFunction	17
		6.4.3.2	applyResult	17
		6.4.3.3	applyResult	17
		6.4.3.4	getNextFunction	18
		6.4.3.5	launch	18
		6.4.3.6	launchAndWait	19
		6.4.3.7	numThreads	19
		6.4.3.8	outputResult	19
		6.4.3.9	wait	19
	6.4.4	Member	Data Documentation	20
		6.4.4.1	m_functions	20
		6.4.4.2	m_futures	20
		6.4.4.3	m_numThreads	20
		6.4.4.4	m_outputResult	20
6.5	policy::	NoPolicy (	Class Reference	20
	6.5.1	Detailed	Description	21
6.6	Result	Policy Clas	ss Reference	22
Eilo	Doouma	entation		25
7.1			x File Reference	25
7.1			zer.hxx File Reference	26
7.2			eference	27
7.3 7.4	•			27
1.4	กษอนให้	UIICY.HXX	File Reference	21

7

# Namespace Index

1	.1	Namespace	List

Here	is a	list o	of all	namespaces	with	brief	descriptions
11010	io a	iioi (	Ji ali	Halliespaces	VVILII	וטווטו	ucscriptions

lactory								٠																9
policy																								9
thread																		 						10

2 Namespace Index

# **Hierarchical Index**

# 2.1 Class Hierarchy

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

policy::AddNumberResultPolicy	1
factory::AsyncFutureFactory	
FutureFactory	3
thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >	4
policy::NoPolicy	)
ResultPolicy	2
thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >	4
OutputResult *	
size_t	
vector< std::function< OutputResult()>>	
vector< std::future< OutputResult > >	

**Hierarchical Index** 

# **Class Index**

## 3.1 Class List

Here are the classes	s, structs,	unions	and inte	rfaces	with	brief	descri	ptions
----------------------	-------------	--------	----------	--------	------	-------	--------	--------

oolicy::AddNumberResultPolicy	
actory::AsyncFutureFactory	
FutureFactory	
$hread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult > \ldots \ldots \ldots$	
policy::NoPolicy	
ResultPolicy	

6 Class Index

# File Index

## 4.1 File List

Here	ic a	list of	all fi	les with	hrief	descriptions
Hele	is a	1151 01	all II	ies willi	Dilei	descriptions

FutureFactory.hxx	25
FutureSynchronyzer.hxx	26
Policy.hxx	27
ResultPolicy.hxx	27

8 File Index

# **Namespace Documentation**

### 5.1 factory Namespace Reference

#### Classes

class AsyncFutureFactory

#### 5.1.1 Detailed Description

The MIT License (MIT)

Copyright (c) <2015> < Romain LEGUAY romain.leguay@gmail.com>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, IN ← CLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

#### 5.2 policy Namespace Reference

#### **Classes**

- · class AddNumberResultPolicy
- · class NoPolicy

#### **Typedefs**

typedef NoPolicy NoResultPolicy

#### 5.2.1 Detailed Description

The MIT License (MIT)

Copyright (c) <2015> < Romain LEGUAY romain.leguay@gmail.com>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, IN ← CLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

#### 5.2.2 Typedef Documentation

#### 5.2.2.1 typedef NoPolicy policy::NoResultPolicy

The class does nothing with the result passed to it's arguments

#### 5.3 thread Namespace Reference

#### **Classes**

· class FutureSynchronyzer

#### 5.3.1 Detailed Description

The MIT License (MIT)

Copyright (c) <2015> <Romain LEGUAY romain.leguay@gmail.com>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, IN ← CLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# **Class Documentation**

### 6.1 policy::AddNumberResultPolicy Class Reference

```
#include <ResultPolicy.hxx>
```

Collaboration diagram for policy::AddNumberResultPolicy:

### policy::AddNumberResultPolicy

- + AddNumberResultPolicy()
- + ~AddNumberResultPolicy()
- + applyResult()

#### **Public Member Functions**

- AddNumberResultPolicy ()
- virtual ~AddNumberResultPolicy ()
- template<typename OutputResult >
   void applyResult (OutputResult \*out, OutputResult \*otherValue)

#### 6.1.1 Detailed Description

This class provides a policy to add a value to another one.

#### 6.1.2 Constructor & Destructor Documentation

- **6.1.2.1** policy::AddNumberResultPolicy::AddNumberResultPolicy( ) [inline]
- **6.1.2.2** virtual policy::AddNumberResultPolicy::~AddNumberResultPolicy( ) [inline], [virtual]

12 Class Documentation

#### 6.1.3 Member Function Documentation

6.1.3.1 template<typename OutputResult > void policy::AddNumberResultPolicy::applyResult ( OutputResult \* out, OutputResult \* otherValue ) [inline]

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

· ResultPolicy.hxx

### 6.2 factory::AsyncFutureFactory Class Reference

```
#include <FutureFactory.hxx>
```

Collaboration diagram for factory::AsyncFutureFactory:

#### factory::AsyncFutureFactory

- + AsyncFutureFactory()
- + ~AsyncFutureFactory()
- + createFuture()

#### **Public Member Functions**

- AsyncFutureFactory ()
- virtual ~AsyncFutureFactory ()
- template < typename OutputResult >
   std::future < OutputResult > createFuture (const std::function < OutputResult() > &f)

#### 6.2.1 Detailed Description

This class provides a factory to build futre with std::async methods.

#### 6.2.2 Constructor & Destructor Documentation

- **6.2.2.1** factory::AsyncFutureFactory::AsyncFutureFactory() [inline]
- **6.2.2.2 virtual factory::**AsyncFutureFactory::~AsyncFutureFactory( ) [inline],[virtual]

#### 6.2.3 Member Function Documentation

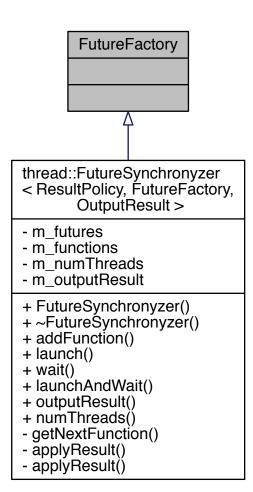
6.2.3.1 template<typename OutputResult > std::future<OutputResult> factory::AsyncFutureFactory::createFuture ( const std::function<OutputResult()> & f ) [inline]

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

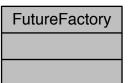
• FutureFactory.hxx

### 6.3 FutureFactory Class Reference

Inheritance diagram for FutureFactory:



Collaboration diagram for FutureFactory:



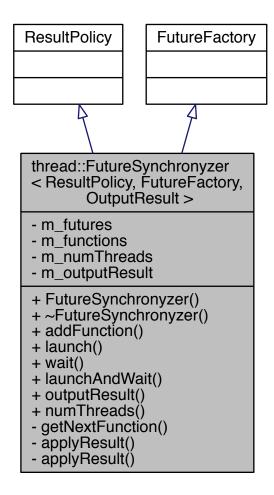
The documentation for this class was generated from the following file:

• FutureSynchronyzer.hxx

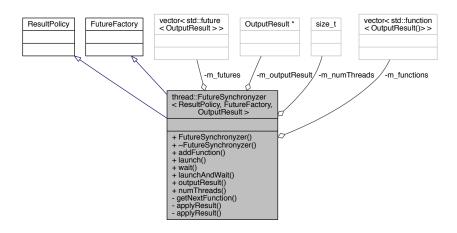
6.4 thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult > Class Template Reference

#include <FutureSynchronyzer.hxx>

Inheritance diagram for thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >:



Collaboration diagram for thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >:



16 Class Documentation

#### **Public Member Functions**

- FutureSynchronyzer (OutputResult \*outputResult=NULL, size\_t numThreads=std::thread::hardware\_← concurrency() > 0?std::thread::hardware concurrency():1)
- virtual ~FutureSynchronyzer ()
- template<typename Fn , typename... Args>
   void addFunction (Fn &&function, Args &&...args)
- · void launch ()
- void wait ()
- void launchAndWait ()
- OutputResult \* outputResult () const
- size t numThreads () const

#### **Private Member Functions**

- std::function < OutputResult() > getNextFunction ()
- template<typename Output = OutputResult, typename Policy = ResultPolicy>
   std::enable\_if<!(std::is\_same< Output, void >::value)&!(std::is\_same< Policy, policy::NoResultPolicy >
   ::value), void >::type applyResult (std::future< Output > &f)
- template<typename Output = OutputResult, typename Policy = ResultPolicy> std::enable\_if<(std::is\_same< Output, void >::value)||(std::is\_same< Policy, policy::NoResultPolicy > ::value), void >::type applyResult (std::future< Output > &f)

#### **Private Attributes**

- std::vector< std::future< OutputResult >> m futures
- std::vector< std::function< OutputResult()>> m\_functions
- size\_t m\_numThreads
- OutputResult \* m\_outputResult

#### 6.4.1 Detailed Description

template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename Output  $\leftarrow$  Result = void > class thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult >

This class permits to add functions and it's arguments and launch them with a limit number of threads defined at the onbject's construction. All the function added must returned the same type of argument.

#### 6.4.2 Constructor & Destructor Documentation

#### Construct the Future Synchronizer.

#### Parameters

outputResult | the return result (optional);

numThreads	the max number of threads. By default, the number of concurrent threads supported if de-	]
	tected or 1.	

- 6.4.2.2 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
  OutputResult = void> virtual thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult
  >::~FutureSynchronyzer() [inline], [virtual]
- 6.4.3 Member Function Documentation
- 6.4.3.1 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
  OutputResult = void> template < typename Fn , typename... Args> void thread::FutureSynchronyzer <
  ResultPolicy, FutureFactory, OutputResult > ::addFunction ( Fn && function, Args &&... args ) [inline]

Add the function and its arguments to the list of function

#### **Parameters**

function	
args	all arguments

6.4.3.2 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename OutputResult = void> template < typename Output = OutputResult, typename Policy = ResultPolicy> std::enable\_if <!(std::is\_same < Output, void>::value) && !(std::is\_same < Policy, policy::NoResultPolicy>::value), void>::type thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult >::applyResult ( std::future < Output > & f ) [inline], [private]

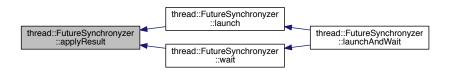
Launch the result operation. This method is used if OutputResult is not void and ResultPolicy is not ResultPolicy.

#### **Parameters**

f	the finished future with the result

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launch(), and thread ::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::wait().

Here is the caller graph for this function:



Launch the result operation. This method is used if OutputResult is void or ResultPolicy is ResultPolicy.

18 Class Documentation

#### **Parameters**

f	the finished future with the result

#### Returns

the next function to be treated.

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launch(). Here is the caller graph for this function:

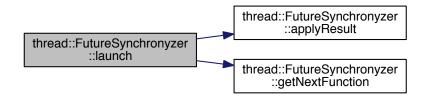


6.4.3.5 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
OutputResult = void> void thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult
>::launch( ) [inline]

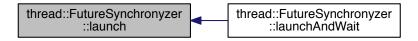
#### Launch all functions

References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::applyResult(), thread  $\leftarrow$  ::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::getNextFunction(), and thread::Future  $\leftarrow$  Synchronyzer< ResultPolicy, FutureFactory, OutputResult >::m\_numThreads.

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launchAndWait(). Here is the call graph for this function:



Here is the caller graph for this function:

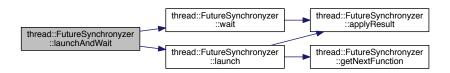


6.4.3.6 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
OutputResult = void > void thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult
>::launchAndWait() [inline]

Launch all the functions and wait them to be finished.

References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launch(), and thread:: $\leftarrow$  FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::wait().

Here is the call graph for this function:



6.4.3.7 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
OutputResult = void > size\_t thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult
>::numThreads ( ) const [inline]

References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::m\_numThreads.

6.4.3.8 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
OutputResult = void > OutputResult \* thread::FutureSynchronyzer < ResultPolicy, FutureFactory,
OutputResult >::outputResult ( ) const [inline]

References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::m outputResult.

6.4.3.9 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
OutputResult = void > void thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult
>::wait ( ) [inline]

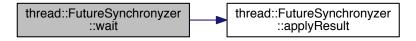
Wait all functions to be finished.

References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::applyResult().

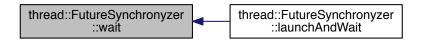
 $Referenced \ by \ thread:: Future Synchronyzer < Result Policy, \ Future Factory, \ Output Result > :: launch And Wait().$ 

20 Class Documentation

Here is the call graph for this function:



Here is the caller graph for this function:



#### 6.4.4 Member Data Documentation

- 6.4.4.1 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
  OutputResult = void> std::vector<std::function<OutputResult()>> thread::FutureSynchronyzer<
  ResultPolicy, FutureFactory, OutputResult>::m\_functions [private]
- 6.4.4.2 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
  OutputResult = void> std::vector < std::future < OutputResult> > thread::FutureSynchronyzer < ResultPolicy,
  FutureFactory, OutputResult >::m futures [private]
- 6.4.4.3 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
  OutputResult = void > size\_t thread::FutureSynchronyzer < ResultPolicy, FutureFactory, OutputResult
  >::m\_numThreads [private]

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launch(), and thread ::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::numThreads().

6.4.4.4 template < class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename
OutputResult = void > OutputResult \* thread::FutureSynchronyzer < ResultPolicy, FutureFactory,
OutputResult >::m\_outputResult [private]

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::outputResult(). The documentation for this class was generated from the following file:

• FutureSynchronyzer.hxx

### 6.5 policy::NoPolicy Class Reference

#include <Policy.hxx>

Collaboration diagram for policy::NoPolicy:

policy::NoPolicy

### 6.5.1 Detailed Description

This class represents an empty policy

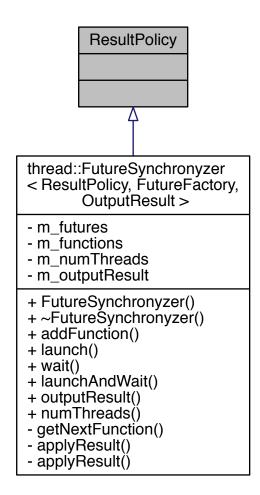
The documentation for this class was generated from the following file:

• Policy.hxx

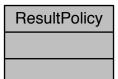
22 Class Documentation

### 6.6 ResultPolicy Class Reference

Inheritance diagram for ResultPolicy:



Collaboration diagram for ResultPolicy:



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

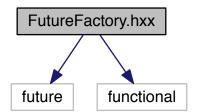
• FutureSynchronyzer.hxx

24 **Class Documentation** 

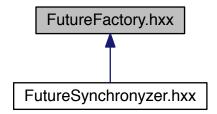
# **File Documentation**

## 7.1 FutureFactory.hxx File Reference

#include <future>
#include <functional>
Include dependency graph for FutureFactory.hxx:



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



26 File Documentation

#### Classes

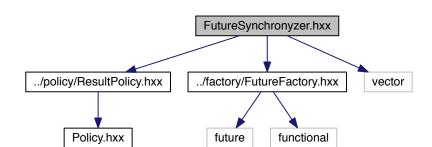
· class factory::AsyncFutureFactory

### **Namespaces**

· factory

## 7.2 FutureSynchronyzer.hxx File Reference

```
#include "../policy/ResultPolicy.hxx"
#include "../factory/FutureFactory.hxx"
#include <vector>
Include dependency graph for FutureSynchronyzer.hxx:
```



#### Classes

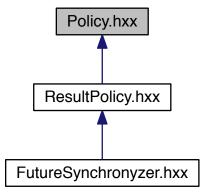
• class thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >

#### **Namespaces**

thread

## 7.3 Policy.hxx File Reference

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



#### Classes

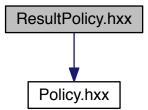
• class policy::NoPolicy

#### **Namespaces**

policy

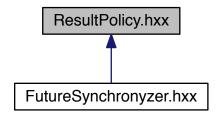
## 7.4 ResultPolicy.hxx File Reference

#include "Policy.hxx"
Include dependency graph for ResultPolicy.hxx:



28 File Documentation

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



#### Classes

• class policy::AddNumberResultPolicy

#### **Namespaces**

policy

### Typedefs

• typedef NoPolicy policy::NoResultPolicy