FutureSynchronyzer

/doc

Generated by Doxygen 1.8.9.1

Wed Feb 25 2015 21:48:59

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

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

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

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

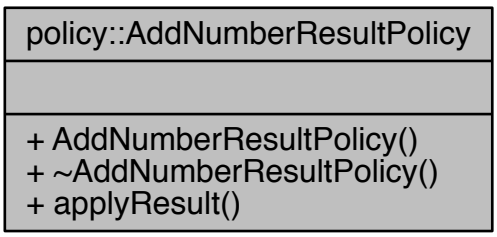
Chapter 6

Class Documentation

6.1 policy::AddNumberResultPolicy Class Reference

`#include <ResultPolicy.hxx>`

Collaboration diagram for policy::AddNumberResultPolicy:



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]

6.1.3 Member Function Documentation

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

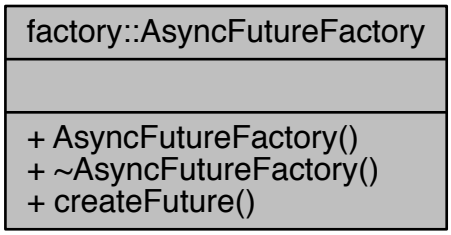
The documentation for this class was generated from the following file:

- ResultPolicy.hxx

6.2 factory::AsyncFutureFactory Class Reference

`#include <FutureFactory.hxx>`

Collaboration diagram for factory::AsyncFutureFactory:



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

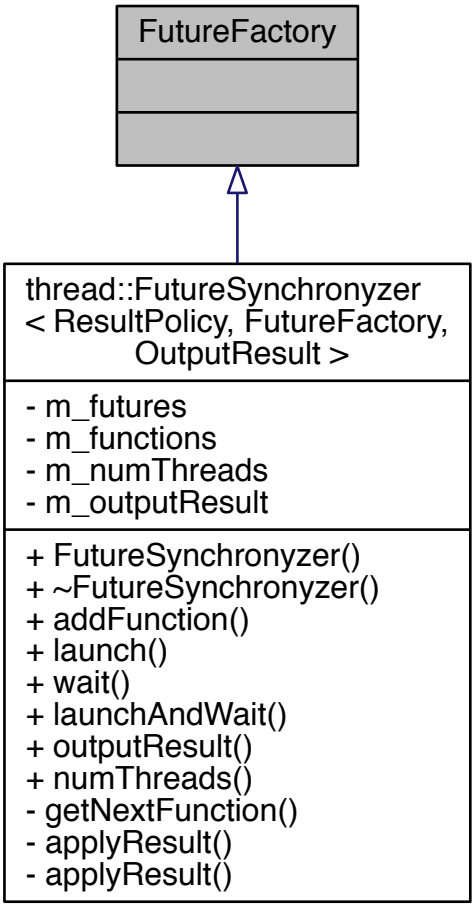
6.2.3.1 template<typename OutputResult > std::future<OutputResult> factory::AsyncFutureFactory::createFuture (const std::function< OutputResult()> & *f*) [inline]

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

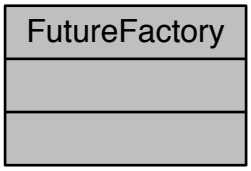
- FutureFactory.hxx

6.3 FutureFactory Class Reference

Inheritance diagram for FutureFactory:



Collaboration diagram for FutureFactory:



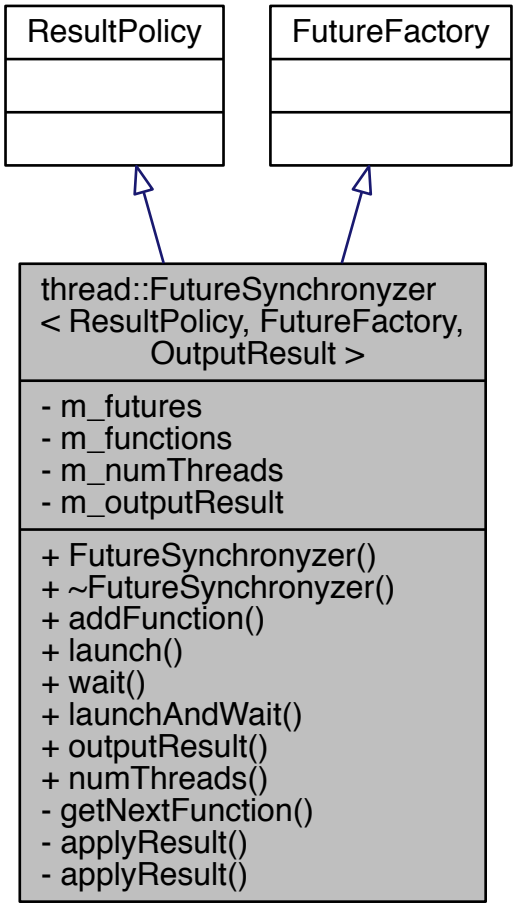
The documentation for this class was generated from the following file:

- FutureSynchronyzer.hxx

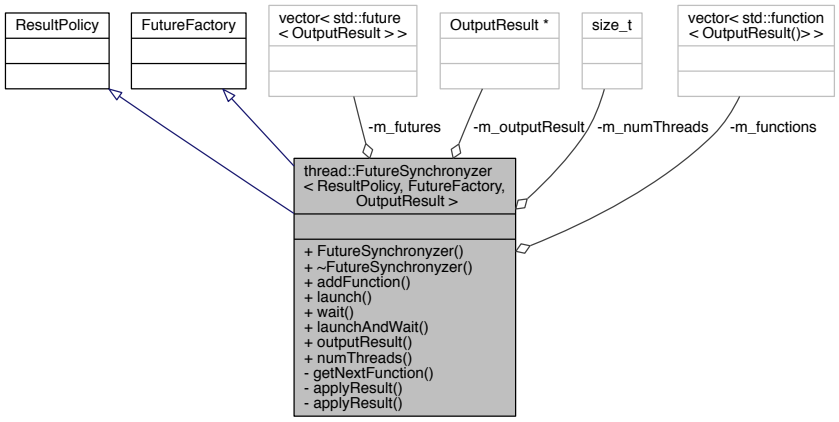
6.4 thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult > Class Template Reference

`#include <FutureSynchronyzer.hxx>`

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



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



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 OutputResult = 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

6.4.2.1 template<class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename OutputResult = void> thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::FutureSynchronyzer (OutputResult * *outputResult* = NULL, size_t *numThreads* = std::thread::hardware_concurrency(), 0?std::thread::hardware_concurrency():1) [inline]

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 detected 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]

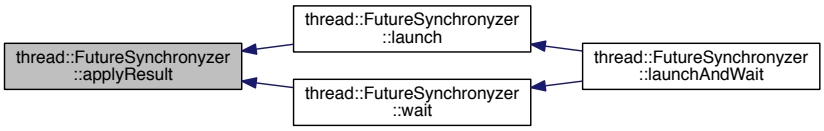
Launch the result operation. This method is used if OutputResult is not void and ResultPolicy is not ResultPolicy.

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:



6.4.3.3 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 void or ResultPolicy is ResultPolicy.

Parameters

f	the finished future with the result

6.4.3.4 template<class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename OutputResult = void> std::function<OutputResult()> thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::getNextFunction ( ) [inline], [private]

Returns

the next function to be treated.

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

Here is the caller graph for this function:

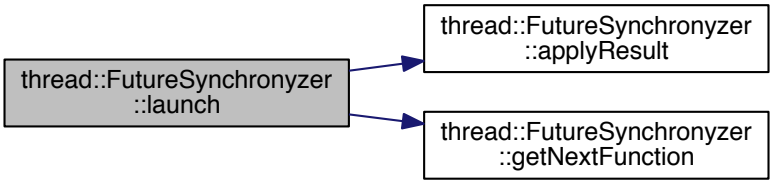
6.4.3.5 template<class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename OutputResult = void> void thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launch ( ) [inline]

Launch all functions

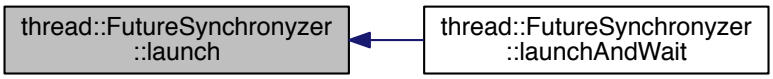
References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::applyResult(), thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::getNextFunction(), and thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::m_numThreads.

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launchAndWait().

Here is the call graph for this function:



Here is the caller graph for this function:

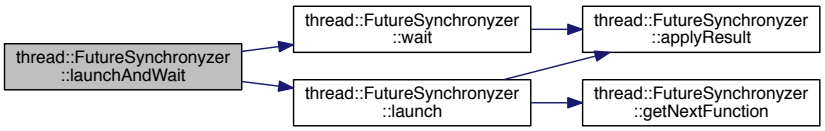


6.4.3.6 template<class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename OutputResult = void> void thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launchAndWait ( ) [inline]

Launch all the functions and wait them to be finished.

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

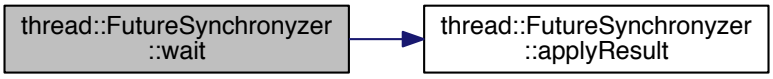
6.4.3.9 template<class ResultPolicy = policy::NoResultPolicy, class FutureFactory = factory::AsyncFutureFactory, typename OutputResult = void> void thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::wait ( ) [inline]

Wait all functions to be finished.

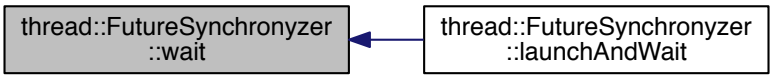
References thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::applyResult().

Referenced by thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >::launchAndWait().

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().

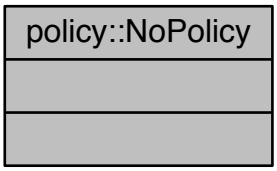
The documentation for this class was generated from the following file:

- FutureSynchronyzer.hxx

6.5 policy::NoPolicy Class Reference

`#include <Policy.hxx>`

Collaboration diagram for policy::NoPolicy:



6.5.1 Detailed Description

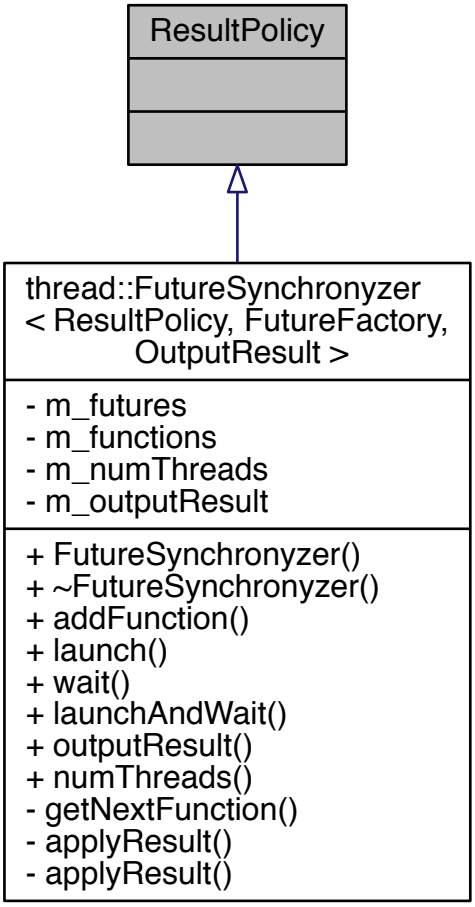
This class represents an empty policy

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

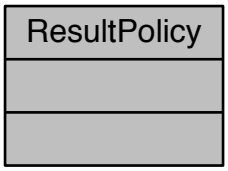
- Policy.hxx

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

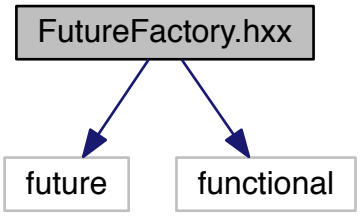
Chapter 7

File Documentation

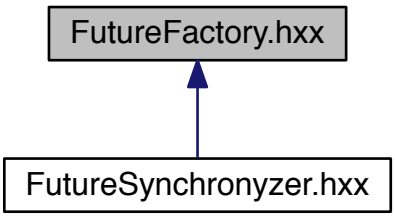
7.1 FutureFactory.hxx File Reference

```
#include <future>
#include <functional>
```

Include dependency graph for FutureFactory.hxx:



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



Classes

- class factory::AsyncFutureFactory

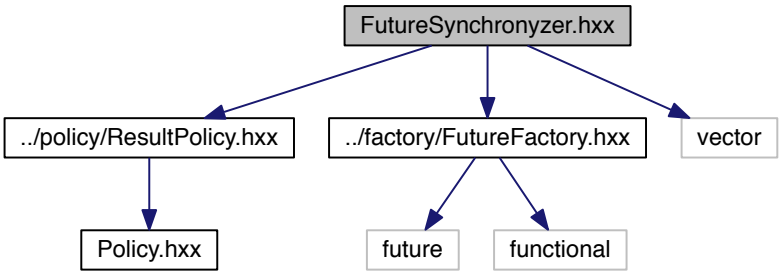
Namespaces

- factory

7.2 FutureSynchronyzer.hxx File Reference

```
#include "../policy/ResultPolicy.hxx"
#include "../factory/FutureFactory.hxx"
#include <vector>
```

Include dependency graph for FutureSynchronyzer.hxx:



Classes

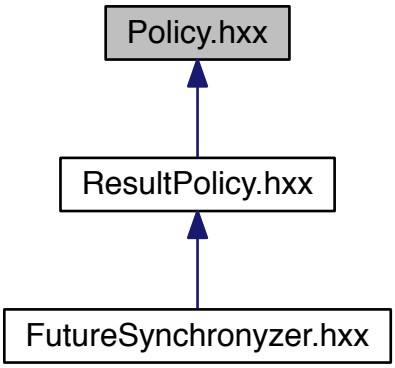
- class thread::FutureSynchronyzer< ResultPolicy, FutureFactory, OutputResult >

Namespaces

- thread

7.3 Policy.hxx File Reference

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



Classes

- class policy::NoPolicy

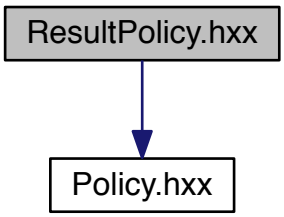
Namespaces

- policy

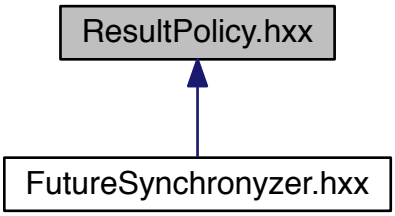
7.4 ResultPolicy.hxx File Reference

```
#include "Policy.hxx"
```

Include dependency graph for ResultPolicy.hxx:



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



Classes

- class policy::AddNumberResultPolicy

Namespaces

- policy

Typedefs

- typedef NoPolicy policy::NoResultPolicy