comode

Generated by Doxygen 1.8.5

Wed Sep 24 2014 05:17:31

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented 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

Namespace Documentation

4.1 coco Namespace Reference

Classes

- class PropertyBase
- class Property

  type spec
- class AttributeBase
- class Attribute

  template spec of attribute
- class OperationBase
- class Operation
- struct ConnectionPolicy
- class ConnectionBase
- class OutputPort

  Class representing an output port containing data of type T.
- class InputPort

  Class representing an input port containing data of type T.
- class ConnectionT

  Template class to manage the type of the connection.
- class ConnectionDataL

  Specialized class for the type T to manage ConnectionPolicy::DATA ConnectionPolicy::LOCKED.
- class ConnectionDataU

  Specialized class for the type T to manage ConnectionPolicy::DATA ConnectionPolicy::UNSYNC.
- class ConnectionBufferU

  Specialized class for the type T to manage ConnectionPolicy::BUFFER/CIRCULAR_BUFFER ConnectionPolicy::UNSYNC.
- class ConnectionBufferL

  Specialized class for the type T to manage ConnectionPolicy::BUFFER/CIRCULAR_BUFFER ConnectionPolicy::LOCKED.
- class ConnectionManager

  Manage the connections of one PortBase.
- class PortBase

  Base class to manage ports.
- class RunnableInterface

  Interface class to execute the components.
- class ExecutionEngine

- struct SchedulePolicy

  policy for executing the component
- class Activity
- class SequentialActivity
- class ParallelActivity
- class Service
- class TaskContext
- class TaskContextT
- class ComponentSpec
- class ComponentRegistry

Enumerations

- enum FlowStatus { **NO_DATA**, **OLD_DATA**, **NEW_DATA** }

  status of a connection port
- enum TaskState { **INIT**, **PRE_OPERATIONAL**, **STOPPED**, **RUNNING** }

  state of a TaskContext
- enum ThreadSpace { **OWN_THREAD**, **CLIENT_THREAD** }

  policy for the execution of operations

Functions

- Activity * **createSequentialActivity** (SchedulePolicy sp, std::shared_ptr< RunnableInterface > r=0)
- Activity * **createParallelActivity** (SchedulePolicy sp, std::shared_ptr< RunnableInterface > r=0)
- template<class T >
  ConnectionT< T > * makeConnection (InputPort< T > *a, OutputPort< T > *b, ConnectionPolicy p)

4.1.1 Detailed Description

Micro Component Framework – Orocos RTT 2.0 inspired C++11

Principles:

- TaskContext
- Port (in and out)
- Property (for params)
- Connection (connecting In and Out port)

Ownership:

- ports are managed by

Progress:

- loading of components

4.1.2 Function Documentation

4.1.2.1 template<class T > ConnectionT< T > * coco::makeConnection (InputPort< T > * *a*, OutputPort< T > * *b*, ConnectionPolicy *p*)

Factory fo the connection policy

References coco::ConnectionPolicy::buffer_size_, and coco::ConnectionPolicy::data_policy_.

Referenced by coco::InputPort< T >::connectToTyped(), and coco::OutputPort< T >::connectToTyped().

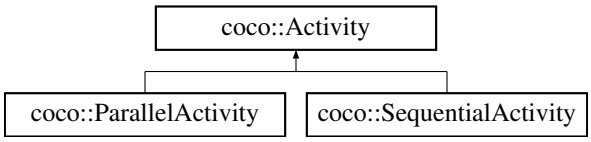
Chapter 5

Class Documentation

5.1 coco::Activity Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::Activity:



Public Member Functions

- Activity (SchedulePolicy policy, std::shared_ptr< RunnableInterface > r)
  specify the execution policy and the RunnableInterface to be executed
- virtual void start ()=0
  Start the activity.
- virtual void stop ()=0
  Stop the activity.
- virtual bool **isPeriodic** ()
- virtual void trigger ()=0
  in case of a TRIGGER activity starts one step of the execution
- virtual void entry ()=0
  main execution function
- void init ()
  initialize the acrivity and the Engine
- bool **isActive** ()
- SchedulePolicy::Policy **getPolicyType** ()

Protected Attributes

- std::shared_ptr
  < RunnableInterface > **runnable_**
- SchedulePolicy **policy_**
- bool **active_**
- std::atomic< bool > **stopping_**

5.1.1 Detailed Description

Base class for something that loops or is activated

TODO: ownership TODO: specialized for: Sequential and Parallel

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

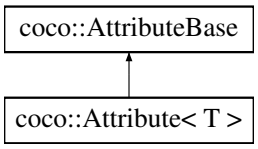
- src/ezoro.hpp

5.2 coco::Attribute< T > Class Template Reference

template spec of attribute

`#include <ezoro.hpp>`

Inheritance diagram for coco::Attribute< T >:



Public Member Functions

- **Attribute** (TaskContext *p, const char *name)

Public Attributes

- T **value**

5.2.1 Detailed Description

template<class T>class coco::Attribute< T >

template spec of attribute

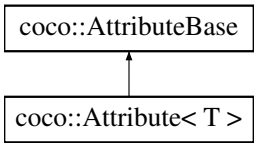
The documentation for this class was generated from the following file:

- src/ezoro.hpp

5.3 coco::AttributeBase Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::AttributeBase:



Public Member Functions

- **AttributeBase** (TaskContext *p, const char *name)

Public Attributes

- std::string **name_**

5.3.1 Detailed Description

run-time value virtual

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

- src/ezoro.hpp
- src/ezoro.cpp

5.4 coco::ComponentRegistry Class Reference

```
#include <ezoro.hpp>
```

Static Public Member Functions

- static TaskContext * create (const char *name)
  creates a component by name
- static void addSpec (ComponentSpec *s)
  adds a specification
- static bool addLibrary (const char *)
  adds a library
- static void alias (const char *newname, const char *oldname)
  defines an alias. note that oldname should be present

Private Member Functions

- TaskContext * **create_** (const char *name)
- void **addSpec_** (ComponentSpec *s)
- bool **addLibrary_** (const char *)
- void **alias_** (const char *newname, const char *oldname)

Static Private Member Functions

- static ComponentRegistry & **get** ()

Private Attributes

- std::map< std::string, ComponentSpec * > **specs**
- std::set< std::string > **libs**
- std::map< std::string, std::shared_ptr< TaskContext > > **contexts**

5.4.1 Detailed Description

Component Registry that is singleton per each exec or library. Then when the component library is loaded the singleton is replacedù

add the addition of a full path to automatically add libraries

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

- src/ezoro.hpp
- src/ezoro.cpp

5.5 coco::ComponentSpec Class Reference

`#include <ezoro.hpp>`

Public Types

- typedef std::function
  < TaskContext ∗()> **makefx_t**

Public Member Functions

- **ComponentSpec** (const char ∗name, makefx_t fx)

Public Attributes

- std::string **name_**
- makefx_t **fx_**

5.5.1 Detailed Description

Specification of the component, as created by the macro

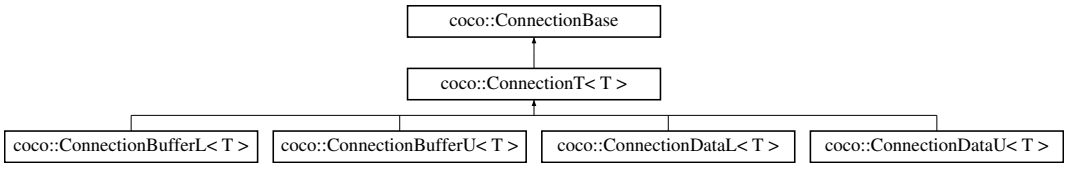
The documentation for this class was generated from the following files:

- src/ezoro.hpp
- src/ezoro.cpp

5.6 coco::ConnectionBase Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::ConnectionBase:



Public Member Functions

- ConnectionBase (PortBase *in, PortBase *out, ConnectionPolicy policy)

  Constructor.
- bool hasNewData () const

Protected Member Functions

- void trigger ()

Protected Attributes

- FlowStatus data_status_

  status of the data in the container
- ConnectionPolicy policy_

  policy for data management
- PortBase * input_ = 0

  input port untyped
- PortBase * output_ = 0

  output port untyped

5.6.1 Detailed Description

Base class for connections

5.6.2 Member Function Documentation

5.6.2.1 bool coco::ConnectionBase::hasNewData ( ) const [inline]

Returns

NEW_DATA if new data is present in the Input port

References data_status_.

5.6.2.2 void coco::ConnectionBase::trigger ( ) [protected]

Trigger the port to communicate new data is present

References input_, and coco::PortBase::triggerComponent().

Referenced by coco::ConnectionDataL< T >::addData(), coco::ConnectionDataU< T >::addData(), coco::ConnectionBufferU< T >::addData(), and coco::ConnectionBufferL< T >::addData().

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

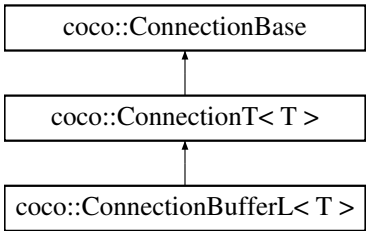
- src/ezoro.hpp
- src/ezoro.cpp

5.7 coco::ConnectionBufferL< T > Class Template Reference

Specialized class for the type T to manage ConnectionPolicy::BUFFER/CIRCULAR_BUFFER ConnectionPolicy::LOCKED.

`#include <ezoro.hpp>`

Inheritance diagram for coco::ConnectionBufferL< T >:



Public Member Functions

- ConnectionBufferL (InputPort< T > *in, OutputPort< T > *out, ConnectionPolicy policy)

  Simply call ConnectionT<T> constructor.
- virtual FlowStatus **getNewestData** (T &data)
- virtual FlowStatus getData (T &data) override

  If there is new data in the container retrive it.
- virtual bool addData (T &input) override

  Add data to the container. If the input port is of type event trigger it to wake up the execution.

Private Attributes

- boost::circular_buffer< T > **buffer_**
- std::mutex **mutex_t_**

Additional Inherited Members

5.7.1 Detailed Description

template<class T>class coco::ConnectionBufferL< T >

Specialized class for the type T to manage ConnectionPolicy::BUFFER/CIRCULAR_BUFFER ConnectionPolicy::LOCKED.

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

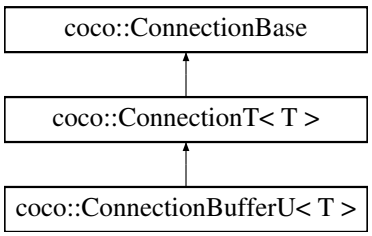
- src/ezoro.hpp

5.8 coco::ConnectionBufferU< T > Class Template Reference

Specialized class for the type T to manage ConnectionPolicy::BUFFER/CIRCULAR_BUFFER ConnectionPolicy::UNSYNC.

`#include <ezoro.hpp>`

Inheritance diagram for coco::ConnectionBufferU< T >:



Public Member Functions

- ConnectionBufferU (InputPort< T > *in, OutputPort< T > *out, ConnectionPolicy policy)

  Simply call ConnectionT<T> constructor.
- virtual FlowStatus **getNewestData** (T &data)
- virtual FlowStatus getData (T &data) override

  If there is new data in the container retrive it.
- virtual bool addData (T &input) override

  Add data to the container. If the input port is of type event trigger it to wake up the execution.

Private Attributes

- boost::circular_buffer< T > **buffer_**

Additional Inherited Members

5.8.1 Detailed Description

template<class T>class coco::ConnectionBufferU< T >

Specialized class for the type T to manage ConnectionPolicy::BUFFER/CIRCULAR_BUFFER ConnectionPolicy::-UNSYNC.

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

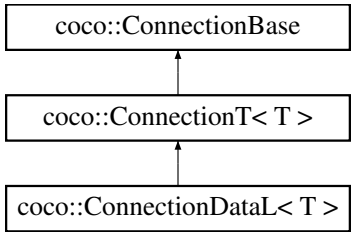
- src/ezoro.hpp

5.9 coco::ConnectionDataL< T > Class Template Reference

Specialized class for the type T to manage ConnectionPolicy::DATA ConnectionPolicy::LOCKED.

`#include <ezoro.hpp>`

Inheritance diagram for coco::ConnectionDataL< T >:



Public Member Functions

- ConnectionDataL (InputPort< T > *in, OutputPort< T > *out, ConnectionPolicy policy)
  Simply call ConnectionT<T> constructor.
- virtual FlowStatus getData (T &data) override
  If there is new data in the container retrive it.
- virtual bool addData (T &input) override
  Add data to the container. If the input port is of type event trigger it to wake up the execution.

Private Attributes

- union {
  T **value_t_**
  };
- std::mutex **mutex_t_**

Additional Inherited Members

5.9.1 Detailed Description

template<class T>class coco::ConnectionDataL< T >

Specialized class for the type T to manage ConnectionPolicy::DATA ConnectionPolicy::LOCKED.

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

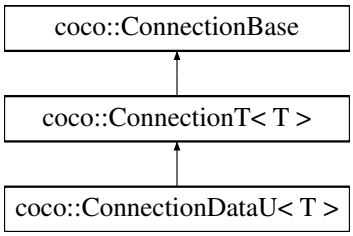
- src/ezoro.hpp

5.10 coco::ConnectionDataU< T > Class Template Reference

Specialized class for the type T to manage ConnectionPolicy::DATA ConnectionPolicy::UNSYNC.

`#include <ezoro.hpp>`

Inheritance diagram for coco::ConnectionDataU< T >:



Public Member Functions

- ConnectionDataU (InputPort< T > *in, OutputPort< T > *out, ConnectionPolicy policy)
  Simply call ConnectionT<T> constructor.
- virtual FlowStatus getData (T &data) override
  If there is new data in the container retrive it.
- virtual bool addData (T &input) override
  Add data to the container. If the input port is of type event trigger it to wake up the execution.

Private Attributes

- union {
  T **value_t_**
  };

Additional Inherited Members

5.10.1 Detailed Description

template<class T>class coco::ConnectionDataU< T >

Specialized class for the type T to manage ConnectionPolicy::DATA ConnectionPolicy::UNSYNC.

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

- src/ezoro.hpp

5.11 coco::ConnectionManager Class Reference

Manage the connections of one PortBase.

`#include <ezoro.hpp>`

Public Member Functions

- ConnectionManager (PortBase *o)

  set the RounRobin index to 0 and set its PortBase ptr owner
- bool addConnection (std::shared_ptr< ConnectionBase > connection)

  add a connection to `connections_`
- bool hasConnections () const

  return true if `connections_` has at list one elemnt
- std::shared_ptr< ConnectionBase > getConnection (int index)

  return the ConnectionBase connection inidicated by index if it exist
- template<class T >
  std::shared_ptr< ConnectionT< T > > getConnectionT (int index)

  return the ConnectionT<T> connection inidicated by index if it exist
- int connectionsSize () const

  return the number of connections

Protected Attributes

- int rr_index_

  round robin index to poll the connection when reading data
- PortBase * owner_

  PortBase pointer owning this manager.
- std::vector< std::shared_ptr
  < ConnectionBase > > connections_

  List of ConnectionBase associate to `owner_`.

Friends

- template<class T >
  class **InputPort**
- template<class T >
  class **OutputPort**

5.11.1 Detailed Description

Manage the connections of one PortBase.

Specialized class for the type T

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

- src/ezoro.hpp

5.12 coco::ConnectionPolicy Struct Reference

`#include <ezoro.hpp>`

Public Types

- enum **Policy** { **DATA**, **BUFFER**, **CIRCULAR** }
- enum **LockPolicy** { **UNSYNC**, **LOCKED**, **LOCK_FREE** }
- enum **Transport** { **LOCAL**, **IPC** }

Public Member Functions

- ConnectionPolicy ()
- **ConnectionPolicy** (Policy policiy, int buffer_size, bool blocking=false)

Public Attributes

- Policy data_policy_
- LockPolicy **lock_policy_**
- int buffer_size_
- bool init_
- std::string **name_id_**
- Transport **transport_** = LOCAL

5.12.1 Detailed Description

Connection Policy

5.12.2 Constructor & Destructor Documentation

5.12.2.1 coco::ConnectionPolicy::ConnectionPolicy ( ) [inline]

Default constructor, default value:

Parameters

*data_policy_*	= DATA
*lock_policy_*	= LOCKED
*buffer_size_*	= 1
*init_*	= 1

5.12.3 Member Data Documentation

5.12.3.1 int coco::ConnectionPolicy::buffer_size_

type of lock policy

Referenced by coco::ConnectionBufferL< T >::ConnectionBufferL(), coco::ConnectionBufferU< T >::ConnectionBufferU(), and coco::makeConnection().

5.12.3.2 Policy coco::ConnectionPolicy::data_policy_

type of data container

Referenced by coco::ConnectionBufferU< T >::addData(), coco::ConnectionBufferL< T >::addData(), and coco::makeConnection().

5.12.3.3 bool coco::ConnectionPolicy::init_

size of the data container

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

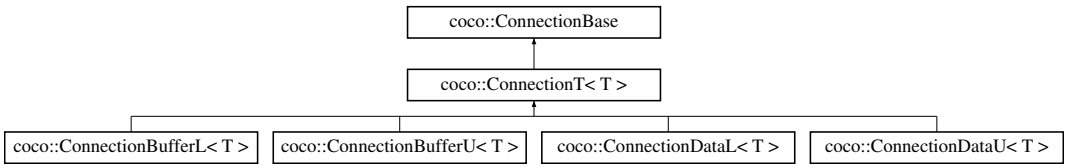
- src/ezoro.hpp

5.13 coco::ConnectionT< T > Class Template Reference

Template class to manage the type of the connection.

`#include <ezoro.hpp>`

Inheritance diagram for coco::ConnectionT< T >:



Public Member Functions

- ConnectionT (InputPort< T > *in, OutputPort< T > *out, ConnectionPolicy policy)

  Simply call ConnectionBase constructor.
- virtual FlowStatus getData (T &data)=0

  If there is new data in the container retrive it.
- virtual bool addData (T &data)=0

  Add data to the container. If the input port is of type event trigger it to wake up the execution.

Additional Inherited Members

5.13.1 Detailed Description

template<class T>class coco::ConnectionT< T >

Template class to manage the type of the connection.

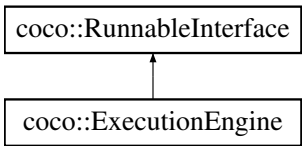
The documentation for this class was generated from the following file:

- src/ezoro.hpp

5.14 coco::ExecutionEngine Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::ExecutionEngine:



Public Member Functions

- ExecutionEngine (TaskContext *t)

  constructor to set the current TaskContext to be executed
- virtual void init () override

  Initialize the components members.
- virtual void step () override

  If the task is running execute uno step of the execution function.
- virtual void finalize () override

  when the task is stopped clear all the members

Protected Attributes

- TaskContext * **task_** = 0
- bool **stopped_**

Additional Inherited Members

5.14.1 Detailed Description

Concrete class to execture the components

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

- src/ezoro.hpp
- src/ezoro.cpp

5.15 coco::impl::getfunctioner< T > Struct Template Reference

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

- src/ezoro.hpp

5.16 coco::impl::getfunctioner< R(Args...)> Struct Template Reference

Public Types

- typedef std::function< R(Args...)> **target**
- using **fx** = R(Args...)

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

- src/ezoro.hpp

5.17 coco::impl::getfunctioner< R(U::*)(Args...) > Struct Template Reference

Public Types

- typedef std::function< R(Args...)> **target**
- using **fx** = R(Args...)

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

- src/ezoro.hpp

5.18 coco::impl::getfunctioner< std::function< R(Args...) > > Struct Template Reference

Public Types

- typedef std::function< R(Args...)> **target**
- using **fx** = R(Args...)

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

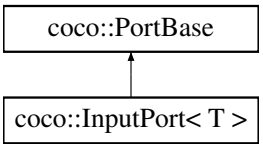
- src/ezoro.hpp

5.19 coco::InputPort< T > Class Template Reference

Class representing an input port containing data of type T.

`#include <ezoro.hpp>`

Inheritance diagram for coco::InputPort< T >:



Public Member Functions

- InputPort (TaskContext *p, const char *name, bool is_event=false)
  simply call PortBase constructor
- const std::type_info & getTypeInfo () const override
  return the template type name of the port data
- bool connectTo (PortBase *other, ConnectionPolicy policy)
  connect a port to another with a specified ConnectionPolicy
- FlowStatus read (T &output)
  using a round robin schedule polls all its connections to see if someone has new data to be read

Private Member Functions

- std::shared_ptr< ConnectionT< T > > getConnection (int index)
  get the connection at position `index`
- bool connectToTyped (OutputPort< T > *other, ConnectionPolicy policy)
  connect the current port with `other`

Additional Inherited Members

5.19.1 Detailed Description

template<class T>class coco::InputPort< T >

Class representing an input port containing data of type T.

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

- src/ezoro.hpp

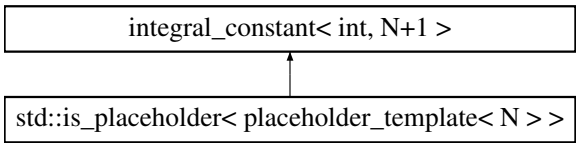
5.20 coco::impl::int_sequence<> Struct Template Reference

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

- src/ezoro.hpp

5.21 std::is_placeholder< placeholder_template< N > > Struct Template Reference

Inheritance diagram for std::is_placeholder< placeholder_template< N > >:



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

- src/ezoro.hpp

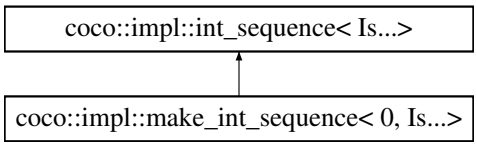
5.22 coco::impl::make_int_sequence< N, Is > Struct Template Reference

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

- src/ezoro.hpp

5.23 coco::impl::make_int_sequence< 0, Is...> Struct Template Reference

Inheritance diagram for coco::impl::make_int_sequence< 0, Is...>:



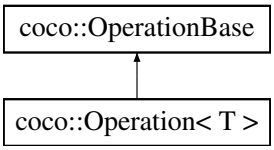
The documentation for this struct was generated from the following file:

- src/ezoro.hpp

5.24 coco::Operation< T > Class Template Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::Operation< T >:



Public Types

- typedef T **value_t**
- typedef coco::impl::getfunctioner< T > ::fx **Sig**

Public Member Functions

- **Operation** (Service *p, const T &fx, const char *name)
- virtual const std::type_info & assig ()

  returns the type signature
- virtual void * asfx ()

  commented for future impl

Public Attributes

- T **fx_**

5.24.1 Detailed Description

template<class T>class coco::Operation< T >

Operator Class specialized for T as Signature

5.24.2 Member Function Documentation

5.24.2.1 template<class T > virtual void* coco::Operation< T >::asfx ( ) [inline], [virtual]

commented for future impl

returns the contained function pointer

Implements coco::OperationBase.

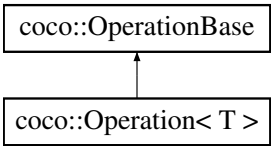
The documentation for this class was generated from the following file:

- src/ezoro.hpp

5.25 coco::OperationBase Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::OperationBase:



Public Member Functions

- **OperationBase** (Service *p, const char *name)
- virtual void * asfx ()=0

  commented for future impl
- virtual const std::type_info & assig ()=0

  returns the type signature
- template<class Sig >
  std::function< Sig > & as ()

  returns the function as Signature if it matches, otherwise raises exception

Public Attributes

- std::string **name_**

5.25.1 Detailed Description

Basic Class for Operations

5.25.2 Member Function Documentation

5.25.2.1 virtual void* coco::OperationBase::asfx ( ) [pure virtual]

commented for future impl

returns the contained function pointer

Implemented in coco::Operation< T >.

Referenced by as().

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

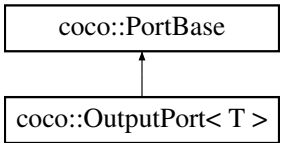
- src/ezoro.hpp

5.26 coco::OutputPort< T > Class Template Reference

Class representing an output port containing data of type T.

`#include <ezoro.hpp>`

Inheritance diagram for coco::OutputPort< T >:



Public Member Functions

- OutputPort (TaskContext *p, const char *name)

  simply call PortBase constructor
- const std::type_info & getTypeInfo () const override

  return the template type name of the port data
- bool connectTo (PortBase *other, ConnectionPolicy policy)

  connect a port to another with a specified ConnectionPolicy
- bool write (T &input)

  writes in each of its connections `input`

Private Member Functions

- std::shared_ptr< ConnectionT< T > > getConnection (int index)

  get the connection at position `index`
- bool connectToTyped (InputPort< T > *other, ConnectionPolicy policy)

  connect the current port with `other`

Additional Inherited Members

5.26.1 Detailed Description

template<class T>class coco::OutputPort< T >

Class representing an output port containing data of type T.

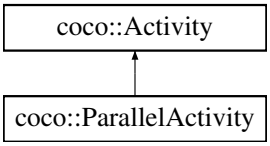
The documentation for this class was generated from the following file:

- src/ezoro.hpp

5.27 coco::ParallelActivity Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::ParallelActivity:



Public Member Functions

- ParallelActivity (SchedulePolicy policy, std::shared_ptr< RunnableInterface > r=nullptr)
  simply call Activity constructor
- virtual void start () override
  Start the activity.
- virtual void stop () override
  Stop the activity.
- virtual void trigger () override
  in case of a TRIGGER activity starts one step of the execution

Protected Member Functions

- void entry () override
  main execution function

Protected Attributes

- std::unique_ptr< std::thread > **thread_**
- std::mutex **mutex_t_**
- std::condition_variable **cond_**

5.27.1 Detailed Description

Uses thread

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

- src/ezoro.hpp
- src/ezoro.cpp

5.28 std::placeholder_template< int > Struct Template Reference

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

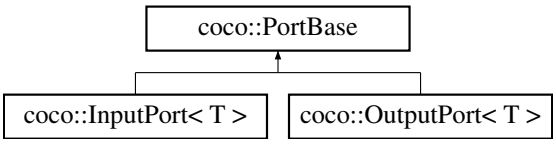
- src/ezoro.hpp

5.29 coco::PortBase Class Reference

Base class to manage ports.

`#include <ezoro.hpp>`

Inheritance diagram for coco::PortBase:



Public Member Functions

- PortBase (TaskContext *p, const char *name, bool is_output, bool is_event)
  initialize input and output ports
- virtual const std::type_info & getTypeInfo () const =0
  return the template type name of the port data
- virtual bool connectTo (PortBase *, ConnectionPolicy policy)=0
  connect a port to another with a specified ConnectionPolicy
- bool isConnected () const
  return true if this port is connected to another one
- bool isEvent () const
  return true if this port is of type event
- void triggerComponent ()

Public Attributes

- std::shared_ptr< TaskContext > task_
  Task using this port.

Protected Member Functions

- bool addConnection (std::shared_ptr< ConnectionBase > connection)
  add a connection to the ConnectionManager
- int connectionsCount () const
  returns the number of connections associate to this port

Protected Attributes

- ConnectionManager **manager_** = { this }
- std::string **name_**
- bool **is_event_**
- bool **is_output_**

Friends

- template<class T >
  class **InputPort**
- template<class T >
  class **OutputPort**

5.29.1 Detailed Description

Base class to manage ports.

5.29.2 Member Function Documentation

5.29.2.1 void coco::PortBase::triggerComponent ( )

Trigger the task to notify new dara is present in the port

References task_.

Referenced by coco::ConnectionBase::trigger().

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

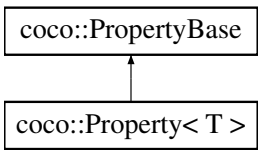
- src/ezoro.hpp
- src/ezoro.cpp

5.30 coco::Property< T > Class Template Reference

type spec

`#include <ezoro.hpp>`

Inheritance diagram for coco::Property< T >:



Public Member Functions

- **Property** (TaskContext *p, const char *name)

Public Attributes

- T **value**

5.30.1 Detailed Description

template<class T>class coco::Property< T >

type spec

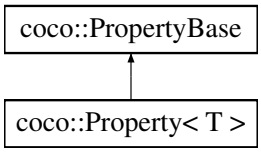
The documentation for this class was generated from the following file:

- src/ezoro.hpp

5.31 coco::PropertyBase Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::PropertyBase:



Public Member Functions

- PropertyBase (TaskContext *p, const char *name)

Public Attributes

- std::string **name_**

5.31.1 Detailed Description

param time virtual

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

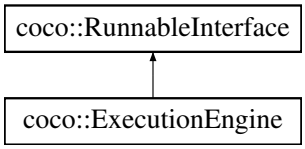
- src/ezoro.hpp
- src/ezoro.cpp

5.32 coco::RunnableInterface Class Reference

Interface class to execute the components.

`#include <ezoro.hpp>`

Inheritance diagram for coco::RunnableInterface:



Public Member Functions

- virtual void init ()=0

  Initialize the components members.
- virtual void step ()=0

If the task is running execute uno step of the execution function.

- virtual void finalize ()=0

  when the task is stopped clear all the members

Public Attributes

- Activity * **activity_** = 0

5.32.1 Detailed Description

Interface class to execute the components.

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

- src/ezoro.hpp

5.33 coco::SchedulePolicy Struct Reference

policy for executing the component

`#include <ezoro.hpp>`

Public Types

- enum **Policy** { **PERIODIC**, **HARD**, **TRIGGERED** }

Public Member Functions

- **SchedulePolicy** (Policy policy=PERIODIC, int period=1)

Public Attributes

- Policy **timing_policy_** = PERIODIC
- int **period_ms_**
- std::string **trigger_**

5.33.1 Detailed Description

policy for executing the component

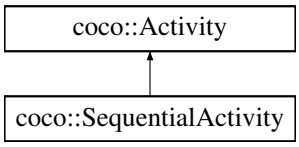
The documentation for this struct was generated from the following file:

- src/ezoro.hpp

5.34 coco::SequentialActivity Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::SequentialActivity:



Public Member Functions

- **SequentialActivity** (SchedulePolicy policy, std::shared_ptr< RunnableInterface > r=nullptr)
- virtual void start () override

  Start the activity.
- virtual void stop () override

  Stop the activity.
- virtual void trigger () override

  in case of a TRIGGER activity starts one step of the execution

Protected Member Functions

- void entry () override

  main execution function

Additional Inherited Members

5.34.1 Detailed Description

No thread but thread safe

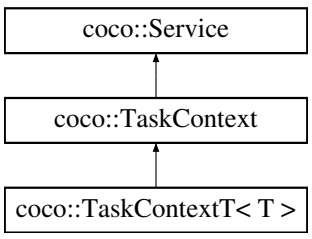
The documentation for this class was generated from the following files:

- src/ezoro.hpp
- src/ezoro.cpp

5.35 coco::Service Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::Service:



Public Member Functions

- Service (const char *n="")

  does nothing

- const PortBase * getPort (std::string name)

  return a port based on its name
- std::list< std::shared_ptr < OperationBase > > & operations ()

  return the list of operations
- template<class T , class Y > void **addOperator** (const char *name, Y b, T a)
- Service * provides ()

  returns self as provider
- Service * provides (const char *x)

  check for sub services

Public Attributes

- std::map< std::string, PortBase * > **ports_**

Private Attributes

- std::string **name_**
- std::list< PropertyBase * > **self_props_**
- std::list< AttributeBase * > **attributes_**
- std::list< std::shared_ptr < OperationBase > > **operations_**
- std::map< std::string, std::unique_ptr< Service > > **subservices**

Friends

- class **PropertyBase**
- class **AttributeBase**
- class **OperationBase**
- class **PortBase**

5.35.1 Detailed Description

Manages all properties of a Task Context. Services is present because Task Context can have sub ones

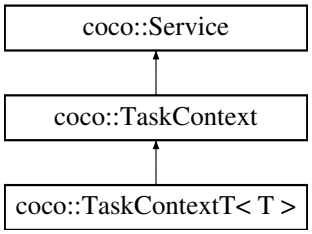
The documentation for this class was generated from the following files:

- src/ezoro.hpp
- src/ezoro.cpp

5.36 coco::TaskContext Class Reference

`#include <ezoro.hpp>`

Inheritance diagram for coco::TaskContext:



Public Member Functions

- void init ()
  init the task
- void setActivity (Activity *activity)
  set the activity that will manage the execution of this task
- void start ()
  start the execution
- void stop ()
  stop the execution of the component
- void triggerActivity ()
  in case of a TRIGGER task execute one step
- virtual const std::type_info & **type** () const =0

Public Attributes

- TaskState **state_**

Protected Member Functions

- TaskContext ()
  creates an ExecutionEngine object
- void prepareUpdate ()
  called every time before executing the component function
- virtual void onConfig ()=0
  function to be overload by the user. It is called in the init phase
- virtual void onUpdate ()=0
  function to be overload by the user. It is the execution funciton

Protected Attributes

- std::shared_ptr< ExecutionEngine > **engine_**

Private Attributes

- Activity * **activity_**
- std::string **name_**

Friends

- class **System**
- class **ExecutionEngine**

5.36.1 Detailed Description

The Task Context is the single task of the Component being instantiated

A Task Context provides:

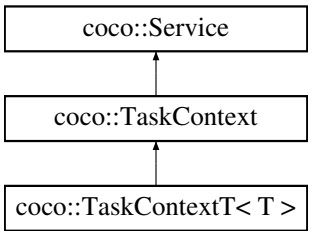
- operators
- input ports
- output ports
- parameters (config time)
- properties (run time)

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

- src/ezoro.hpp
- src/ezoro.cpp

5.37 coco::TaskContextT< T > Class Template Reference

Inheritance diagram for coco::TaskContextT< T >:



Public Member Functions

- virtual const std::type_info & **type** () const override

Additional Inherited Members

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

- src/ezoro.hpp

Index