Class IOSpec

Table of contents

- Defined in File io_spec.hpp

Class Documentation

class IOSpec

Class to define the specification of an input/output port of an Operator.

An interaction point between two operators. Operators ingest data at Input ports and publish data at Output ports. Receiver, Transmitter, and MessageRouter in GXF would be replaced with the concept of Input/Output Port of the Operator and the Flow (Edge) of the Application Workflow in the Framework.

Public Types

enum class IOType

Input/Output type.

Values:

enumerator kInput

enumerator kOutput

enum class ConnectorType

Connector type. Determines the type of Receiver (when IOType is kInput) or Transmitter (when IOType is kOutput) class used.

Values:

enumerator kDefault

enumerator kDoubleBuffer

enumerator kUCX

Public Functions

inline IOSpec(OperatorSpec *op_spec, const std::string &name, IOType io_type)

Construct a new IOSpec object.

Parameters

- **op_spec** – The pointer to the operator specification that contains this input/output.
- **name** – The name of this input/output.
- **io_type** – The type of this input/output.

inline IOSpec(OperatorSpec *op_spec, const std::string &name, IOType io_type, const std::type_info *typeinfo)

Construct a new IOSpec object.

Parameters

- **op_spec** – The pointer to the operator specification that contains this input/output.
- **name** – The name of this input/output.
- **io_type** – The type of this input/output.
- **typeinfo** – The type info of the data of this input/output.

inline OperatorSpec *op_spec() const

Get the operator specification that contains this input/output.

Returns

The pointer to the operator specification that contains this input/output.

inline const std::string &name() const

Get the name of this input/output.

Returns

The name of this input/output.

inline IOType io_type() const

Get the input/output type.

Returns

The input/output type.

inline ConnectorType connector_type() const

Get the receiver/transmitter type.

Returns

The receiver type (for inputs) or transmitter type (for outputs)

inline const std::type_info *typeinfo() const

Get the type info of the data of this input/output.

Returns

The type info of the data of this input/output.

inline std::vector<std::pair<ConditionType, std::shared_ptr<Condition>>>
&conditions()

Get the conditions of this input/output.

Returns

The reference to the conditions of this input/output.

template<typename ...ArgsT>
inline IOSpec &condition(ConditionType type, ArgsT&&... args)

Add a condition to this input/output.

The following ConditionTypes are supported:

- ConditionType::kMessageAvailable

- ConditionType::kDownstreamAffordable
- ConditionType::kNone

Parameters

- **type** – The type of the condition.
- **args** – The arguments of the condition.

Returns

The reference to this IOSpec.

inline std::shared_ptr<Resource> connector() const

Get the connector (transmitter or receiver) of this input/output.

Returns

The connector (transmitter or receiver) of this input/output.

inline void connector(std::shared_ptr<Resource> connector)

Set the connector (transmitter or receiver) of this input/output.

Parameters

connector – The connector (transmitter or receiver) of this input/output.

template<typename ...ArgsT>
inline IOSpec &connector(ConnectorType type, ArgsT&&... args)

Add a connector (receiver/transmitter) to this input/output.

The following ConnectorTypes are supported:

- ConnectorType::kDefault
- ConnectorType::kDoubleBuffer
- ConnectorType::kUCX

Parameters

- **type** – The type of the connector (receiver/transmitter).
- **args** – The arguments of the connector (receiver/transmitter).

Returns

The reference to this IOSpec.

virtual YAML::Node to_yaml_node() const

Get a YAML representation of the IOSpec.

Returns

YAML node including the parameters of this component.

std::string description() const

Get a description of the IOSpec.

to_yaml_node()

Returns

YAML string.