Class CodecRegistry

Table of contents

- Defined in File codec_registry.hpp

Class Documentation

class CodecRegistry

Class to set codecs for data types.

This class is used to set codecs (serializer + deserializer) for data types.

Public Types

using GXFEndpoint = nvidia::gxf::Endpoint

using SerializeFunc = std::function<nvidia::gxf::Expected<size_t>(const Message&, GXFEndpoint*)>

Function type for serializing a data type.

using DeserializeFunc = std::function<nvidia::gxf::Expected<Message>(GXFEndpoint*)>

Function type for deserializing a data type.

using Codec = std::pair<SerializeFunc, DeserializeFunc>

Function pair type for serialization and deserialization of a data type.

Public Functions

inline Codec &get_codec(const std::type_index &index)

Get the codec function pair.

Parameters

index – The type index of the parameter.

Returns

The reference to the Codec object.

inline Codec &get_codec(const std::string &codec_name)

Get the codec function pair.

Parameters

codec_name – The name of the codec.

Returns

The reference to the Codec object.

inline SerializeFunc &get_serializer(const std::string &codec_name)

Get the serializer function.

Parameters

codec_name – The name of the codec.

Returns

The reference to the Serializer function.

inline SerializeFunc &get_serializer(const std::type_index &index)

Get the serializer function.

Parameters

index – The type index of the parameter.

Returns

The reference to the Serializer function.

inline DeserializeFunc &get_deserializer(const std::string &codec_name)

Get the deserializer function.

Parameters

codec_name – The name of the codec.

Returns

The reference to the Deserializer function.

inline DeserializeFunc &get_deserializer(const std::type_index &index)

Get the deserializer function.

Parameters

index – The type index of the parameter.

Returns

The reference to the Deserializer function.

inline expected<std::type_index, RuntimeError> name_to_index(const std::string &codec_name)

Get the std::type_index corresponding to a codec name.

Parameters

codec_name – The name of the codec.

Returns

The std::type_index corresponding to the name.

inline expected<std::string, RuntimeError> index_to_name(const std::type_index &index)

Get the codec name corresponding to a std::type_index.

Parameters

index – The std::type_index corresponding to the parameter.

Returns

The name of the codec.

template<typename typeT>
inline void add_codec(std::pair<SerializeFunc, DeserializeFunc> codec, const std::string &codec_name, bool overwrite = true)

Add a codec for the type.

Template Parameters

typeT – the type of the parameter.

Parameters

- **codec** – a pair containing a serialization function and deserialization function.
- **codec_name** – the name of the codec to add.
- **overwrite** – if true, any existing codec with matching codec_name will be overwritten.

inline void add_codec(const std::type_index &index, std::pair<SerializeFunc, DeserializeFunc> codec, const std::string &codec_name, bool overwrite = true)

Add a codec for the type.

Parameters

- **index** – the type index of the parameter.
- **codec** – a pair containing a serialization function and deserialization function.
- **codec_name** – the name of the codec to add.
- **overwrite** – if true, any existing codec with matching codec_name will be overwritten.

template<typename typeT>
inline void add_codec(const std::string &codec_name, bool overwrite = true)

Add a codec for the type.

Template Parameters

typeT – the type for which a codec is being added

Parameters

- **index** – the type index of the parameter.
- **codec** – a pair containing a serialization function and deserialization function.
- **codec_name** – The name of the codec to add.
- **overwrite** – if true, any existing codec with matching codec_name will be overwritten.

Public Static Functions

static CodecRegistry &get_instance()

Get the instance object.

Returns

The reference to the CodecRegistry instance.

static inline nvidia::gxf::Expected<size_t> serialize(const Message &message, GXFEndpoint *gxf_endpoint)

Serialize the message object.

Template Parameters

typeT – The data type within the message.

Parameters

- **message** – The message to serialize.
- **endpoint** – The serialization endpoint (buffer).

template<typename typeT>
static inline nvidia::gxf::Expected<Message> deserialize(GXFEndpoint *gxf_endpoint)

Deserialize the message object.

Template Parameters

typeT – The data type within the message.

Parameters

- **message** – The message to serialize.
- **endpoint** – The serialization endpoint (buffer).

Public Static Attributes

static SerializeFunc none_serialize =
```
[]([[maybe_unused]] constMessage& message,[[maybe_unused]] GXFEndpoint* buffer) -&gt; nvidia::gxf::Expected&lt;size_t&gt; {HOLOSCAN_LOG_ERROR("Unable toserializemessage");return 0;}
```

static DeserializeFunc none_deserialize =
```
[]([[maybe_unused]] GXFEndpoint* buffer) -&gt; nvidia::gxf::Expected&lt;Message&gt; {HOLOSCAN_LOG_ERROR("Unable todeserializemessage");returnMessage();}
```

static Codec none_codec = std::make_pair(none_serialize, none_deserialize)

Default Codec for Arg.