Program Listing for File message_available.hpp

Return to documentation for file (include/holoscan/core/conditions/gxf/message_available.hpp)

/* * SPDX-FileCopyrightText: Copyright (c) 2022-2024 NVIDIA CORPORATION & AFFILIATES. All rights reserved. * SPDX-License-Identifier: Apache-2.0 * * Licensed under the Apache License, Version 2.0 (the "License"); * you may not use this file except in compliance with the License. * You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License. */ #ifndef HOLOSCAN_CORE_CONDITIONS_GXF_MESSAGE_AVAILABLE_HPP #define HOLOSCAN_CORE_CONDITIONS_GXF_MESSAGE_AVAILABLE_HPP #include <memory> #include "../../gxf/gxf_condition.hpp" namespace holoscan { class MessageAvailableCondition : public gxf::GXFCondition { public: HOLOSCAN_CONDITION_FORWARD_ARGS_SUPER(MessageAvailableCondition, GXFCondition) MessageAvailableCondition() = default; explicit MessageAvailableCondition(size_t min_size) : min_size_(min_size) {} MessageAvailableCondition(size_t min_size, size_t front_stage_max_size) : min_size_(min_size), front_stage_max_size_(front_stage_max_size) {} const char* gxf_typename() const override { return "nvidia::gxf::MessageAvailableSchedulingTerm"; } void receiver(std::shared_ptr<gxf::GXFResource> receiver) { receiver_ = receiver; } std::shared_ptr<gxf::GXFResource> receiver() { return receiver_.get(); } void min_size(uint64_t min_size); size_t min_size() { return min_size_; } void front_stage_max_size(size_t front_stage_max_size); size_t front_stage_max_size() { return front_stage_max_size_; } void setup(ComponentSpec& spec) override; void initialize() override { GXFCondition::initialize(); } nvidia::gxf::MessageAvailableSchedulingTerm* get() const; *// TODO(GXF4): Expected<void> setReceiver(Handle<Receiver> value)* private: *// TODO(GXF4): this is now a std::set<Handle<Receiver>> receivers_* Parameter<std::shared_ptr<gxf::GXFResource>> receiver_; Parameter<uint64_t>

```
min_size_; Parameter<size_t> front_stage_max_size_; }; } // namespace holoscan
#endif/* HOLOSCAN_CORE_CONDITIONS_GXF_MESSAGE_AVAILABLE_HPP */
```