Program Listing for File asynchronous.hpp

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

```
/* * SPDX-FileCopyrightText: Copyright (c) 2023-2024 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_ASYNCHRONOUS_HPP #define HOLOSCAN_CORE_CONDITIONS_GXF_ASYNCHRONOUS_HPP #include <string> #include <gxf/std/scheduling_terms.hpp> #include "../../gxf/gxf_condition.hpp" namespace holoscan { using nvidia::gxf::AsynchronousEventState; class AsynchronousCondition : public gxf::GXFCondition { public: HOLOSCAN_CONDITION_FORWARD_ARGS_SUPER(AsynchronousCondition, GXFCondition) AsynchronousCondition() = default; AsynchronousCondition(const std::string& name, nvidia::gxf::AsynchronousSchedulingTerm* term); const char* gxf_typename() const override { return "nvidia::gxf::AsynchronousSchedulingTerm"; } void setup(ComponentSpec& spec) override; void event_state(AsynchronousEventState state); AsynchronousEventState event_state() const; nvidia::gxf::AsynchronousSchedulingTerm* get() const; private: AsynchronousEventState event_state_{AsynchronousEventState::READY}; }; } // namespace holoscan #endif/* HOLOSCAN_CORE_CONDITIONS_GXF_ASYNCHRONOUS_HPP */
```