Program Listing for File aja_source.hpp

Return to documentation for file (include/holoscan/operators/aja_source/aja_source.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_OPERATORS_AJA_SOURCE_AJA_SOURCE_HPP #define HOLOSCAN_OPERATORS_AJA_SOURCE_AJA_SOURCE_HPP #include <ajantv2/includes/ntv2card.h> #include <ajantv2/includes/ntv2devicescanner.h> #include <ajantv2/includes/ntv2enums.h> #include <string> #include <utility> #include <vector> #include "holoscan/core/io_context.hpp" #include "holoscan/core/io_spec.hpp" #include "holoscan/core/operator.hpp" #include "holoscan/core/operator_spec.hpp" #include "./ntv2channel.hpp" namespace holoscan::ops { class AJASourceOp : public holoscan::Operator { public: HOLOSCAN_OPERATOR_FORWARD_ARGS(AJASourceOp) AJASourceOp(); void setup(OperatorSpec& spec) override; void initialize() override; void start() override; void compute(InputContext& op_input, OutputContext& op_output, ExecutionContext& context) override; void stop() override; private: AJAStatus DetermineVideoFormat(); AJAStatus OpenDevice(); AJAStatus SetupVideo(); AJAStatus SetupBuffers(); AJAStatus StartAutoCirculate(); bool AllocateBuffers(std::vector<void*>& buffers, size_t num_buffers, size_t buffer_size, bool rdma); void FreeBuffers(std::vector<void*>& buffers, bool rdma); bool GetNTV2VideoFormatTSI(NTV2VideoFormat* format); Parameter<holoscan::IOSpec*> video_buffer_output_; Parameter<std::string> device_specifier_; Parameter<NTV2Channel> channel_; Parameter<uint32_t> width_; Parameter<uint32_t> height_; Parameter<uint32_t> framerate_; Parameter<bool> use_rdma_; Parameter<bool> enable_overlay_; Parameter<NTV2Channel> overlay_channel_; Parameter<bool> overlay_rdma_; Parameter<holoscan::IOSpec*> overlay_buffer_input_; Parameter<holoscan::IOSpec*> overlay_buffer_output_; //
```

internal state CNTV2Card device_; NTV2DeviceID device_id_; NTV2VideoFormat video_format_; NTV2PixelFormat pixel_format_ = NTV2_FBF_ABGR; bool use_tsi_ = false; bool is_kona_hdmi_ = false; std::vector<void*> buffers_; std::vector<void*> overlay_buffers_; uint8_t current_buffer_ = 0; uint8_t current_hw_frame_ = 0; uint8_t current_overlay_hw_frame_ = 0; bool is_igpu_ = false; }; } *// namespace holoscan::ops* #endif/* HOLOSCAN_OPERATORS_AJA_SOURCE_AJA_SOURCE_HPP */