Program Listing for File holoinfer_constants.hpp

Return to documentation for file (modules/holoinfer/src/include/holoinfer_constants.hpp)

```
/* * SPDX-FileCopyrightText: Copyright (c) 2022-2023 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_INFER_CONSTANTS_H #define _HOLOSCAN_INFER_CONSTANTS_H #include <cuda_runtime_api.h> #include <sys/stat.h> #include <algorithm> #include <fstream> #include <iostream> #include <iterator> #include <map> #include <memory> #include <mutex> #include <numeric> #include <string> #include <utility> #include <vector> #include <holoscan/logger/logger.hpp> #define _HOLOSCAN_EXTERNAL_API_ __attribute__((visibility("default"))) namespace holoscan { namespace inference { enum class holoinfer_datatype { h_Float32 = 0, h_Int8 = 1, h_Int32 = 2, h_Int64 = 3, h_UInt8 = 4, h_Unsupported = 5 }; enum class holoinfer_data_processor { h_CUDA = 0, h_HOST = 1, h_CUDA_AND_HOST = 2 }; enum class holoinfer_backend { h_trt = 0, h_onnx = 1, h_torch = 2, h_unknown = 3 }; enum class holoinfer_code { H_SUCCESS, H_ERROR, H_EXCEPTION, H_WARNING }; class _HOLOSCAN_EXTERNAL_API_ InferStatus { holoinfer_code _code; std::string _message; public: holoinfer_code get_code() const { return _code; } std::string get_message() const { return _message; } void set_code(const holoinfer_code& _c) { _code = _c; } void set_message(const std::string& _m) { _message = _m; } void display_message() const { switch (_code) { case holoinfer_code::H_SUCCESS: default: { HOLOSCAN_LOG_INFO(_message); break; } case holoinfer_code::H_ERROR: { HOLOSCAN_LOG_ERROR(_message); break; } } } InferStatus(const holoinfer_code& code = holoinfer_code::H_SUCCESS, const std::string& message = "") : _code(code), _message(message) {} }; using TimePoint = std::chrono::steady_clock::time_point; using byte = unsigned char; } // namespace inference } // namespace holoscan #endif
```