



## **Program Listing for File `std_component_serializer.hpp`**

[Return to documentation for file \(](#)

`include/holoscan/core/resources/gxf/std_component_serializer.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_RESOURCES_GXF_STD_COMPONENT_SERIALIZER_HPP #define
HOLOSCAN_CORE_RESOURCES_GXF_STD_COMPONENT_SERIALIZER_HPP #include
<memory> #include <vector> // TODO: provide get() method once upstream issue with
missing GXF header is resolved // #include
<gxf/serialization/std_component_serializer.hpp> #include "../gxf/gxf_resource.hpp"
#include "../unbounded_allocator.hpp" namespace holoscan { class
StdComponentSerializer : public gxf::GXFResource { public:
HOLOSCAN_RESOURCE_FORWARD_ARGS_SUPER(StdComponentSerializer,
GXFResource) StdComponentSerializer() = default; const char* gxf_typename() const
override { return "nvidia::gxf::StdComponentSerializer"; } void
setup(ComponentSpec& spec) override; void initialize() override; //
nvidia::gxf::StdComponentSerializer* get() const; private:
Parameter<std::shared_ptr<holoscan::Allocator>> allocator_; }; } // namespace
holoscan #endif/*
HOLOSCAN_CORE_RESOURCES_GXF_STD_COMPONENT_SERIALIZER_HPP */
```

© Copyright 2022-2024, NVIDIA.. PDF Generated on 06/06/2024