**imageNeusoft Confidential**

File No. : ProjectNo.- D00-D01-T01-流水号

NeuSAR aCore

软件需求规格说明书

（Software requirement specification）

**(LT)**

Neusoft Reach Automotive Technology Co., Ltd

Change Log

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Version** | **Contents Revised** | **Status** | **Reviser/**  **Date** | **Approve/Date** |
| 1 |  | 新建 | Draft | 2022.3.25 |  |
| 2 |  | 评审问题修改 | In Review | 2022.4.6 |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Contents**

[1 引言（Introduction） 1](#_Toc100154274)

[1.1 目的（Goal） 1](#_Toc100154275)

[1.2 范围（Scope） 1](#_Toc100154276)

[1.3 参考文档（Reference） 1](#_Toc100154277)

[1.4 术语及缩略语（Terms And Abbreviations） 1](#_Toc100154278)

[2 软件系统概述（Software System Overview） 2](#_Toc100154279)

[2.1 软件系统背景（Software System Background） 2](#_Toc100154280)

[2.2 软件系统目标（Software System Goal） 2](#_Toc100154281)

[2.3 外部关联 （External Association） 2](#_Toc100154282)

[3 功能需求（Functional Requirement） 2](#_Toc100154283)

[3.1 日志和追踪 2](#_Toc100154284)

[3.1.1 [SWRD\_LT\_00001] LT功能概述 2](#_Toc100154285)

[3.2 LT模块启用 2](#_Toc100154286)

[3.2.1 [SWRD\_LT\_00002] 日志缓存 3](#_Toc100154287)

[3.2.2 [SWRD\_LT\_00003] 异常丢弃 3](#_Toc100154288)

[3.2.3 [SWRD\_LT\_00004] 初始化Logging框架 3](#_Toc100154289)

[3.3 创建日志上下文实例 4](#_Toc100154290)

[3.3.1 [SWRD\_LT\_00005] 创建日志上下文 4](#_Toc100154291)

[3.4 记录日志消息 5](#_Toc100154292)

[3.4.1 [SWRD\_LT\_00006] 日志输出模式 5](#_Toc100154293)

[3.4.2 [SWRD\_LT\_00007] 检查日志级别 5](#_Toc100154294)

[3.4.3 [SWRD\_LT\_00008] 支持调整运行时日志级别 6](#_Toc100154295)

[3.4.4 [SWRD\_LT\_00009] 支持非建模与建模类型信息 6](#_Toc100154296)

[3.4.5 [SWRD\_LT\_00010] 记录具有级别的日志信息 6](#_Toc100154297)

[3.4.6 [SWRD\_LT\_00011] Logger::WithLevel 7](#_Toc100154298)

[3.4.7[SWRD\_LT\_00012] 消息格式 7](#_Toc100154299)

[3.4.8[SWRD\_LT\_00013] 记录日志文件大小和数量 8](#_Toc100154300)

[3.5 数据转换功能 8](#_Toc100154301)

[3.5.1 [SWRD\_LT\_00014] 支持数据转换 8](#_Toc100154302)

[3.6 时间戳 8](#_Toc100154303)

[3.6.1 [SWRD\_LT\_00015] LT能够访问同步时间基 8](#_Toc100154304)

[3.6.2 [SWRD\_LT\_00016] 传输时间戳信息 9](#_Toc100154305)

[3.6.3 [SWRD\_LT\_00017] 输出关于本地时间基的信息 9](#_Toc100154306)

[3.6.4 [SWRD\_LT\_00018] 输出关于全局时间基的信息 10](#_Toc100154307)

[3.7 客户端状态 10](#_Toc100154308)

[3.7.1 [SWRD\_LT\_00019] 客户端连接 10](#_Toc100154309)

[4 非功能需求(Non-Functional Requirements) 10](#_Toc100154310)

[5 接口说明（API） 10](#_Toc100154311)

[5.1 接口头文件（API Header files） 10](#_Toc100154312)

[5.2 接口共同数据类型（API Common Data Types） 11](#_Toc100154313)

[5.2.1 [SWRD\_API\_LT\_00001] LogLevel 11](#_Toc100154314)

[5.2.2 [SWRD\_API\_LT\_00002] LogMode 11](#_Toc100154315)

[5.2.3 [SWRD\_API\_LT\_00003] LogHex8 12](#_Toc100154316)

[5.2.4 [SWRD\_API\_LT\_00004] LogHex16 13](#_Toc100154317)

[5.2.5 [SWRD\_API\_LT\_00005] LogHex32 13](#_Toc100154318)

[5.2.6 [SWRD\_API\_LT\_00006] LogHex64 14](#_Toc100154319)

[5.2.7 [SWRD\_API\_LT\_00007] LogBin8 15](#_Toc100154320)

[5.2.8 [SWRD\_API\_LT\_00008] LogBin16 15](#_Toc100154321)

[5.2.9 [SWRD\_API\_LT\_00009] LogBin32 16](#_Toc100154322)

[5.2.10 [SWRD\_API\_LT\_00010] LogBin64 17](#_Toc100154323)

[5.2.11 [SWRD\_API\_LT\_00011] Fmt 17](#_Toc100154324)

[5.2.12 [SWRD\_API\_LT\_00012] Format 18](#_Toc100154325)

[5.2.13 [SWRD\_API\_LT\_00013] ClientState 19](#_Toc100154326)

[5.3 接口定义（API Reference） 20](#_Toc100154327)

[5.3.1 [SWRD\_API\_LT\_00014] CreateLogger 20](#_Toc100154328)

[5.3.2 [SWRD\_API\_LT\_00015] HexFormat(uint8) 20](#_Toc100154329)

[5.3.3 [SWRD\_API\_LT\_00016] HexFormat(int8) 21](#_Toc100154330)

[5.3.4 [SWRD\_API\_LT\_00017] HexFormat(uint16) 22](#_Toc100154331)

[5.3.5 [SWRD\_API\_LT\_00018] HexFormat(int16) 23](#_Toc100154332)

[5.3.6 [SWRD\_API\_LT\_00019] HexFormat(uint32) 24](#_Toc100154333)

[5.3.7 [SWRD\_API\_LT\_00020] HexFormat(int32) 25](#_Toc100154334)

[5.3.8 [SWRD\_API\_LT\_00021] HexFormat(uint64) 26](#_Toc100154335)

[5.3.9 [SWRD\_API\_LT\_00022] HexFormat(int64) 26](#_Toc100154336)

[5.3.10 [SWRD\_API\_LT\_00023] BinFormat(uint8) 27](#_Toc100154337)

[5.3.11 [SWRD\_API\_LT\_00024] BinFormat(int8) 28](#_Toc100154338)

[5.3.12 [SWRD\_API\_LT\_00025] BinFormat(uint16) 29](#_Toc100154339)

[5.3.13 [SWRD\_API\_LT\_00026] BinFormat(int16) 30](#_Toc100154340)

[5.3.14 [SWRD\_API\_LT\_00027] BinFormat(uint32) 31](#_Toc100154341)

[5.3.15 [SWRD\_API\_LT\_00028] BinFormat(int32) 31](#_Toc100154342)

[5.3.16 [SWRD\_API\_LT\_00029] BinFormat(uint64) 32](#_Toc100154343)

[5.3.17 [SWRD\_API\_LT\_00030] BinFormat(int64) 33](#_Toc100154344)

[5.3.18 [SWRD\_API\_LT\_00031] Wrapper object creator 34](#_Toc100154345)

[5.3.19 [SWRD\_API\_LT\_00032] Logger of an argument with attributes 35](#_Toc100154346)

[5.3.20 [SWRD\_API\_LT\_00033] Class LogStream 36](#_Toc100154347)

[5.3.21 [SWRD\_API\_LT\_00034] LogStream::Flush 36](#_Toc100154348)

[5.3.22 [SWRD\_API\_LT\_00035] operator<< (uint8\_t value) 37](#_Toc100154349)

[5.3.23 [SWRD\_API\_LT\_00036] operator<<(bool value) 38](#_Toc100154350)

[5.3.24 [SWRD\_API\_LT\_00037] operator<< (uint16\_t value) 39](#_Toc100154351)

[5.3.25 [SWRD\_API\_LT\_00038] operator<< (uint32\_t value) 40](#_Toc100154352)

[5.3.26 [SWRD\_API\_LT\_00039] operator<< (uint64\_t value) 40](#_Toc100154353)

[5.3.27 [SWRD\_API\_LT\_00040] operator<< (int8\_t value) 41](#_Toc100154354)

[5.3.28 [SWRD\_API\_LT\_00041] operator<< (int16\_t value) 42](#_Toc100154355)

[5.3.29 [SWRD\_API\_LT\_00042] operator<< (int32\_t value) 42](#_Toc100154356)

[5.3.30 [SWRD\_API\_LT\_00043] operator<< (int64\_t value) 43](#_Toc100154357)

[5.3.31 [SWRD\_API\_LT\_00044] operator<<(float value) 44](#_Toc100154358)

[5.3.32 [SWRD\_API\_LT\_00045] operator<<(double value) 45](#_Toc100154359)

[5.3.33 [SWRD\_API\_LT\_00046] operator<<(const LogHex8 &value) 45](#_Toc100154360)

[5.3.34 [SWRD\_API\_LT\_00047] operator<<(const LogHex16 &value) 46](#_Toc100154361)

[5.3.35 [SWRD\_API\_LT\_00048] operator<<(const LogHex32 &value) 47](#_Toc100154362)

[5.3.36 [SWRD\_API\_LT\_00049] operator<<(const LogHex64 &value) 47](#_Toc100154363)

[5.3.37 [SWRD\_API\_LT\_00050] operator<<(const LogBin8 &value) 48](#_Toc100154364)

[5.3.38 [SWRD\_API\_LT\_00051] operator<<(const LogBin16 &value) 49](#_Toc100154365)

[5.3.39 [SWRD\_API\_LT\_00052] operator<<(const LogBin32 &value) 50](#_Toc100154366)

[5.3.40 [SWRD\_API\_LT\_00053] operator<<(const LogBin64 &value) 50](#_Toc100154367)

[5.3.41 [SWRD\_API\_LT\_00054] operator<<(const ara::core::StringView value) 51](#_Toc100154368)

[5.3.42 [SWRD\_API\_LT\_00055] operator<<(const char \*const value) 52](#_Toc100154369)

[5.3.43 [SWRD\_API\_LT\_00056] operator<<(LogStream &out, LogLevel value) 53](#_Toc100154370)

[5.3.44 [SWRD\_API\_LT\_00057] operator<<(LogStream &out, const core::ErrorCode &ec) 53](#_Toc100154371)

[5.3.45 [SWRD\_API\_LT\_00058] operator<<(LogStream &out, const std::chrono::duration< Rep, Period > &value) 54](#_Toc100154372)

[5.3.46 [SWRD\_API\_LT\_00059] operator<<(LogStream &out, const ara::core::InstanceSpecifier &value) 55](#_Toc100154373)

[5.3.47 [SWRD\_API\_LT\_00060] operator<<(LogStream &out, const void \*const value) 56](#_Toc100154374)

[5.3.48 [SWRD\_API\_LT\_00061] operator<<( core::Span< const core::Byte > data) 56](#_Toc100154375)

[5.3.49 [SWRD\_API\_LT\_00062] WithLocation(core::StringView file, int line) 57](#_Toc100154376)

[5.3.50 [SWRD\_API\_LT\_00063] Logger::LogFatal 58](#_Toc100154377)

[5.3.51 [SWRD\_API\_LT\_00064] Logger::LogError 59](#_Toc100154378)

[5.3.52 [SWRD\_API\_LT\_00065] Logger::LogWarn 60](#_Toc100154379)

[5.3.53 [SWRD\_API\_LT\_00066] Logger::LogInfo 60](#_Toc100154380)

[5.3.54 [SWRD\_API\_LT\_00067] Logger::LogDebug 61](#_Toc100154381)

[5.3.55 [SWRD\_API\_LT\_00068] Logger::LogVerbose 62](#_Toc100154382)

[5.3.56 [SWRD\_API\_LT\_00069] Logger::IsEnabled 63](#_Toc100154383)

[5.3.57 [SWRD\_API\_LT\_00070] Logger::WithLevel 63](#_Toc100154384)

[5.3.58 [SWRD\_API\_LT\_00071] RegisterConnectionStateHandler(std::function< void(ClientState)> callback) 64](#_Toc100154385)

[5.3.59 [SWRD\_API\_LT\_00072] Dflt() 65](#_Toc100154386)

[5.3.60 [SWRD\_API\_LT\_00073] Hex() 65](#_Toc100154387)

[5.3.61 [SWRD\_API\_LT\_00074] Hex(std::uint16\_t precision) 66](#_Toc100154388)

[5.3.62 [SWRD\_API\_LT\_00075] Dec() 67](#_Toc100154389)

[5.3.63 [SWRD\_API\_LT\_00076] Dec(std::uint16\_t precision) 67](#_Toc100154390)

[5.3.64 [SWRD\_API\_LT\_00077] Oct() 68](#_Toc100154391)

[5.3.65 [SWRD\_API\_LT\_00078] Oct(std::uint16\_t precision) 68](#_Toc100154392)

[5.3.66 [SWRD\_API\_LT\_00079] Bin() 69](#_Toc100154393)

[5.3.67 [SWRD\_API\_LT\_00080] Bin(std::uint16\_t precision) 70](#_Toc100154394)

[5.3.68 [SWRD\_API\_LT\_00081] DecFloat(std::uint16\_t precision=6) 70](#_Toc100154395)

[5.3.69 [SWRD\_API\_LT\_00082] DecFloatMax() 71](#_Toc100154396)

[5.3.70 [SWRD\_API\_LT\_00083] EngFloat(std::uint16\_t precision=6) 71](#_Toc100154397)

[5.3.71 [SWRD\_API\_LT\_00084] EngFloatMax() 72](#_Toc100154398)

[5.3.72 [SWRD\_API\_LT\_00085] HexFloat(std::uint16\_t precision) 73](#_Toc100154399)

[5.3.73 [SWRD\_API\_LT\_00086] HexFloatMax() 73](#_Toc100154400)

[5.3.74 [SWRD\_API\_LT\_00087] AutoFloat(std::uint16\_t precision=6) 74](#_Toc100154401)

[5.3.75 [SWRD\_API\_LT\_00088] AutoFloatMax() 74](#_Toc100154402)

[附录A- 信息定义 75](#_Toc100154403)

[附录B- 配置信息 77](#_Toc100154404)

# 引言（Introduction）

## 目的（Goal）

文档主要用于对NeuSAR aCore平台产品项目中LT模块进行设计说明。该文档旨在产品平台的长期模块化维护以及面向不同产品项目的快速开发

## 范围（Scope）

本文使用者： LT模块开发人员以及测试人员、Team Leader及产品负责人。

本文使用方法：

对于开发人员，根据本文进行LT模块的代码实现。

对于测试人员，通过理解本文，进行测试用例的制作和后续测试实施。

对于Team Leader及产品负责人，通过理解本文，对本模块的设计进行规范化管理。

## 参考文档（Reference）

|  |  |  |
| --- | --- | --- |
| **序号（No.）** | **文档名（Document Name）** | **版本名（Revision）** |
| 1 | AUTOSAR\_SWS\_LogAndTrace | R-2111 |
| 2 | AUTOSAR\_RS\_LogAndTrace | R-2111 |
| 3 | AUTOSAR\_TPS\_ManifestSpecification | R-2111 |
|  |  |  |
|  |  |  |
|  |  |  |

## 术语及缩略语（Terms And Abbreviations）

|  |  |
| --- | --- |
| **术语**  **Term/Abbreviation** | **描述（Description）** |
| Log and Trace | The official Functional Cluster name that manages the logging |
| L&T | Acronym for Log and Trace |
| LT protocol | Original name of the protocol itself (Log and Trace), specified in the PRS document |
| Logging API | The main logging interface towards user applications as a library |
| Logging back-end | Implementation of the LT protocol, e.g. DLT |
| Logging Client | An external tool which can remotely interact with the Logging frame work |
| Logging framework | Implementation of the software solution used for logging purposes |
| Logging instance | The class that enables the logging functionality and handles a single logging context |
| Log message | Log message, including message header(s) |
| Log severity level | Meta information about the severity of a passed logging information |
| DLT | Diagnostics Log and Trace-a GENIVI Log and Trace daemon implementation of the LT protocol |
| Application process | An executable instance (process) that is running on a Machine |

# 软件系统概述（Software System Overview）

## 软件系统背景（Software System Background）

日志和跟踪为自适应应用程序提供接口，以便将日志信息转发到通信总线、控制台或文件系统。所提供的每个日志信息都有自己的严重性级别。对于每个严重性级别，都提供了一个单独的方法供应用程序或自适应平台服务使用，例如ara::com。此外，还提供了实用方法来将十进制值转换为十六进制数字系统或二进制数字系统。

## 软件系统目标（Software System Goal）

日志和跟踪模块开发的目标是根据AUTOSAR规范，为自适应应用程序或自适应服务提供接口，以便将日志信息转发到通信总线、控制台或文件系统。

## 外部关联 （External Association）

为了将提供的日志信息打包成标准化的交付和表示格式，需要一个协议。为此，可以使用在AUTOSAR联盟内标准化的LT协议。

# 功能需求（Functional Requirement）

## 日志和追踪

### [SWRD\_LT\_00001] LT功能概述

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00001 |
| **Type** | Valid |
| **Description** | LT为自适应应用程序提供接口，以便将日志信息转发到通信总线、控制台或文件系统。所提供的每个日志信息都有自己的严重性级别。对于每个严重性级别，都提供了一个单独的方法供应用程序或自适应平台服务使用。同时提供将十进制值转换为十六进制数字系统或二进制数字系统的方法。 |
| **Upstream ID** | - |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 无 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

## LT模块启用

LT应支持初始化日志框架和注册日志信息的来源。

### [SWRD\_LT\_00002] 日志缓存

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00002 |
| **Type** | Valid |
| **Description** | 在Logging框架能够处理它们之前(例如，守护进程通信未建立)所记录的日志消息将被排队。是否开启早起日志功能由LogAndTraceInstantiation.DltLogSink.bufferOutput来确定，队列大小由LogAndTraceInstantiation. DltLogSink.queueSize定义。如果超过这个大小，最早的条目将被丢弃。 |
| **Upstream ID** | SWS\_LOG\_00001[2111M]、TPS\_MANI\_03162 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 上位机配置LogAndTraceInstantiation. DltLogSink.queueSize，在通信建立连接之前记录日志，Process启动一段时间后，建立与dlt的通信，通过dlt viewer查看，是否有通信建立前的日志。 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 修改 |

### [SWRD\_LT\_00003] 异常丢弃

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00003 |
| **Type** | Valid |
| **Description** | 如果在Logging框架或底层系统中发生任何错误，它的目的是不打扰应用程序进程，并静默地丢弃函数调用。为此，相关接口既不指定返回值也不抛出异常。 |
| **Upstream ID** | SWS\_LOG\_00002 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 制造异常 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | M |
| * **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### [SWRD\_LT\_00004] 初始化Logging框架

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00004 |
| **Type** | Valid |
| **Description** | 在使用任何ara::log API之前，必须初始化Logging框架（通过ara::core::Initialize），否则将导致不确定的行为。应用程序执行清单应该提供以下信息来初始化Logging框架:  —唯一的应用ID  —应用描述  —默认的日志级别  —日志模式  —目录路径(仅当LogMode::kFile作为日志模式给出时需要) |
| **Upstream ID** | SWS\_LOG\_00004、SWS\_LOG\_00003、TPS\_MANI\_03162、TPS\_MANI\_03274[2111A]、TPS\_MANI\_03160[2111M]、TPS\_MANI\_01272、constr\_3426、constr\_3427 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 在上位机LogAndTraceInstance.DltLogSink配置模式、DltLogSinkToPortPrototypeMapping配置DltLogSink输出器与app的映射 、DltApplicationToProcessMapping中配置app与进程的映射关系，检查生成的配置文件中是否有上述的信息。在ara::core::Initialize之前调用CreateLogger，使用返回的上下文的引用记录信息，是否有日志输出。 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 修改 |

## 创建日志上下文实例

需要创建日志上下文实例记录消息。

### [SWRD\_LT\_00005] 创建日志上下文

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00005 |
| **Type** | Valid |
| **Description** | 使用CreateLogger()函数，在Logging框架内部创建一个日志上下文，并将其作为引用返回给正在使用的应用程序。在处理日志消息之前，至少应该有一个日志上下文可用。通过调用CreateLogger()创建日志上下文，需要提供以下参数:  —上下文ID  —上下文描述  —日志级别（可选，默认是LogLevel::kWarn） |
| **Upstream ID** | SWS\_LOG\_00005、SWS\_LOG\_00006 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 完成框架初始后，调用CreateLogger()，使用返回的日志上下文实例记录其日志级别范围内的日志，根据配置的日志模式，看是否有这些日志输出, 同时观察输出日志的头部信息（上下文ID）是否和参数对应 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

## 记录日志消息

应用程序应能够向LT模块发送日志消息，每一条日志消息都具有一个日志级别。

### [SWRD\_LT\_00006] 日志输出模式

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00006 |
| **Type** | Valid |
| **Description** | 日志有三种输出的模式，分别位控制台模式、文件模式、网络模式。其中控制台模式、文件模式下日志的输出不依赖于LT 后端。支持组合配置多个模式，文件和网络模式不能同时配置。 |
| **Upstream ID** | SWS\_LOG\_00019 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 上位机分别配置不同的日志模式，观察日志的输出的模式是否和配置相同；如果是控制台或文件模式，kill掉dlt进程，启动应用程序进程，观察日志是否产生 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### [SWRD\_LT\_00007] 检查日志级别

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00007 |
| **Type** | Valid |
| **Description** | 应用程序进程应该能够通过IsEnabled()检查是否配置了所需的日志严重性级别。这种机制节省了在准备日志信息期间使用的CPU和内存资源，因为稍后Logging框架会对日志信息进行过滤。 |
| **Upstream ID** | SWS\_LOG\_00007、SWS\_LOG\_00070 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 框架初始后，输入某个日志级别运行CreateLogger()，然后调用IsEnabled()，此时会在函数中也会输入一个日志级别，如果不在能够记录日志的级别范围内，会返回false，否则返回true。 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### [SWRD\_LT\_00008] 支持调整运行时日志级别

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00008 |
| **Type** | Valid |
| **Description** | 日志客户端（例如DLT Viewer）可以对应用程序范围的日志级别在运行时调整。 |
| **Upstream ID** | - |
| **Dependencies** | - |
| **Verification Criteria** | 测试 |
| **Verification Criteria** | 以Console或Network模式初始化Logging框架，启动Application process，日志客户端接收日志，在运行期间客户端调整日志级别 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### [SWRD\_LT\_00009] 支持非建模与建模类型信息

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00009 |
| **Type** | Valid |
| **Description** | 支持非建模与建模类型信息，LT记录的信息默认是非建模类型 |
| **Upstream ID** | SWS\_LOG\_00204 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 观察日志头部信息，如“2021/04/22 15:48:51.330510 806215446 224 ECU1 TTLT ABCD log info V 4 ”第10个字段是否为“V”，“V”表示非建模消息类型，N表示建模消息类型查看负载输出情况 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | M |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### [SWRD\_LT\_00010] 记录具有级别的日志信息

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00010 |
| **Type** | Valid |
| **Description** | 每一条记录的日志信息都具有日志级别，LT提供LogFatal、LogError、LogWarn、LogInfo、LogDebug、LogVerbose方法初始化具有日志级别的日志信息，这些API会返回一个临时LogStream对象，通过插入流操作符<<来传递参数。 |
| **Upstream ID** | SWS\_LOG\_00008、SWS\_LOG\_00009、SWS\_LOG\_00010 SWS\_LOG\_00011、SWS\_LOG\_00012、SWS\_LOG\_00013 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | CreateLogger()后，使用上述的API进行日志记录，查看日志输出情况，侧重点：日志级别。 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### [SWRD\_LT\_00011] Logger::WithLevel

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00011 |
| **Type** | Valid |
| **Description** | 要编写带有编程方式确定的日志级别的日志消息，必须调用API Logger::WithLevel(LogLevel logLevel)。 |
| **Upstream ID** | SWS\_LOG\_00130[2111A] |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 创建某级别的日志上下文实例，调用WithLevel，输入日志级别高于创建的日志实例，使用“<<”追加参数，观察日志的输出 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | M |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 新增 |

### 3.4.7[SWRD\_LT\_00012] 消息格式

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00012 |
| **Type** | Valid |
| **Description** | 每一条日志按一定的格式输出：  2019/07/05 09:46:31.416284 30471395 003 ECU1 fusi ctId log info V 2 [ some raw data: 48 65 6c 6c 6f 20 77 6f 72 6c 64 00 00 00 00 00 00 00 00 00]  各种字段含义：时间、消息时间戳、本上下文中的消息计数、ecuId、appId、contextId、消息类型是log、日志级别、是否是非建模模式（V/N）、参数个数（段数）、log内容。 |
| **Upstream ID** | - |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 查看日志输出格式 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

### 3.4.8[SWRD\_LT\_00013] 记录日志文件大小和数量

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00013 |
| **Type** | Valid |
| **Description** | 文件模式下，LT可以控制记录日志的文件大小和文件数量 |
| **Upstream ID** | - |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 文件模式下，查看日志文件。 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

## 数据转换功能

提供将十进制值转换为十六进制或二进制值的方法

### [SWRD\_LT\_00014] 支持数据转换

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00014 |
| **Type** | Valid |
| **Description** | 提供专门的转换API函数，用于将十进制数转换为十六进制或二进制表示的字符串，包括正数和负数，其中负数的最高有效位应设置为1 |
| **Upstream ID** | SWS\_LOG\_00120[2111M]、SWS\_LOG\_00015[2111M]、SWS\_LOG\_00016[2111M]、SWS\_LOG\_00017[2111M] |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 使用专门的转换函数对十进制数进行转换，观察结果是否正确 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | H |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 不变 |

## 时间戳

LT输出的每一条日志信息的头部带有时间戳信息。

### [SWRD\_LT\_00015] LT能够访问同步时间基

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00015 |
| **Type** | Valid |
| **Description** | 日志和跟踪应该可以访问同步的时基。 LogAndTraceInstantiation中的timeBaseResource属性应用于标识时间基。 |
| **Upstream ID** | SWS\_LOG\_00082、TPS\_MANI\_03162 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 上位机LogAndTraceInstantiation引用不同类型的timeBaseResource，查看日志头部的时间戳 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 新增 |

### [SWRD\_LT\_00016] 传输时间戳信息

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00016 |
| **Type** | Valid |
| **Description** | 如果清单配置中没有日志和跟踪模块引用的时间基，则不应传输任何时间戳信息。 |
| **Upstream ID** | SWS\_LOG\_00082、SWS\_LOG\_00083、TPS\_MANI\_03162 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 上位机LogAndTraceInstantiation不引用timeBaseResource，查看日志头部的时间戳 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 新增 |

### [SWRD\_LT\_00017] 输出关于本地时间基的信息

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00017 |
| **Type** | Invalid[TS模块2011不再支持本地时间基] |
| **Description** | 当CreateLogger()函数被调用时，日志和跟踪应该发送一条消息，如果使用的时间基是本地时间基，则发送“local time base used”。 |
| **Upstream ID** | SWS\_LOG\_00091、TPS\_MANI\_03162 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 查看LT模块发送的日志消息 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 新增 |

### [SWRD\_LT\_00018] 输出关于全局时间基的信息

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00018 |
| **Type** | Valid |
| **Description** | 当CreateLogger()函数被调用时，日志和跟踪应该发送一条消息，如果使用的时间基是全局时间基，则发送“global time base used”。 |
| **Upstream ID** | SWS\_LOG\_00091、TPS\_MANI\_03162 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 上位机LogAndTraceInstantiation引用全局的时间基，CreateLogger()调用后，查看是否有这条信息 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 新增 |

## 客户端状态

### [SWRD\_LT\_00019] 客户端连接

|  |  |
| --- | --- |
| **SWRD\_ID** | SWRD\_LT\_00019 |
| **Type** | Valid |
| **Description** | 当ara::core::Deinitialize()被调用后，如果连接了客户端，Logging 框架应注意可以收集缓冲区中的所有剩余消息，同时确保不会有新的客户端建立连接 |
| **Upstream ID** | SWS\_LOG\_00122、SWS\_LOG\_00123 |
| **Dependencies** | - |
| **Verification method** | 测试 |
| **Verification Criteria** | 当ara::core::Deinitialize()被调用后，使用dlt-viewer进行连接查看是否可以建立连接，同时查看已经建立的连接是否还有日志的输出 |
| **ASIL** | QM |
| **Status** | In review |
| **Priority** | N/A |
| **CR** | - |
| **Risk** | 无 |
| **Change Type** | 新增 |

# 非功能需求(Non-Functional Requirements)

无

# 接口说明（API）

## 接口头文件（API Header files）

头文件：logger.h、log\_stream.h、logmanager.h、common.h

## 接口共同数据类型（API Common Data Types）

### [SWRD\_API\_LT\_00001] LogLevel

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00001 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00018 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | enumeration | |
| ***Symbol*** | LogLevel | |
| **Scope:** | namespace ara::log | |
| **Underlying type:** | uint8\_t | |
| **Syntax:** | enum class LogLevel : uint8\_t {...}; | |
| **Values:** | kOff= 0x00 | 无日志记录。 |
| kFatal= 0x01 | 致命错误，无法恢复。 |
| kError= 0x02 | 影响正确功能时出错。 |
| kWarn= 0x03 | 警告如果无法确保正确的行为。 |
| kInfo= 0x04 | 信息，高层次的理解。 |
| kDebug= 0x05 | 程序员能理解的详细信息。 |
| kVerbose= 0x06 | 额外详细的调试消息。 |
| **Header file:** | #include "ara/log/common.h" | |
| **Description:** | 可能的严重性级别列表 | |

### [SWRD\_API\_LT\_00002] LogMode

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00002 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00019 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | enumeration | |
| **Symbol** | LogMode | |
| **Scope:** | namespace ara::log | |
| **Underlying type:** | uint8\_t | |
| **Syntax:** | enum class LogMode : uint8\_t {...}; | |
| **Values:** | kRemote= 0x01 | 远程发送。 |
| kFile= 0x02 | 保存到文件。 |
| kConsole= 0x04 | 转发到控制台输出。 |
| **Header file:** | #include "ara/log/common.h" | |
| **Description:** | 记录模式。标志，用于配置日志消息的接收器。 | |
| **Notes:** | 为了组合标志，至少需要为此类型提供|和&运算符。 | |

### [SWRD\_API\_LT\_00003] LogHex8

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00003 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00108 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogHex8 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogHex8 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint8\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示8位十六进制值数据类型。 | |
| **Additional:** | uint8\_t num\_uint8\_t = 15;  LogHex8{num\_uint8\_t}; | |

### [SWRD\_API\_LT\_00004] LogHex16

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00004 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00109 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogHex16 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogHex16 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint16\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示16位十六进制值数据类型。 | |
| **Additional:** | uint16\_t num\_uint16\_t = 315;  LogHex16{num\_uint16\_t}; | |

### [SWRD\_API\_LT\_00005] LogHex32

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00005 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00110 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogHex64 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogHex32 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint32\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示32位十六进制值数据类型。 | |
| **Additional:** | uint32\_t num\_uint32\_t = 315;  LogHex32{num\_uint32\_t}; | |

### [SWRD\_API\_LT\_00006] LogHex64

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00006 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00111 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogHex64 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogHex64 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint64\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示64位十六进制值数据类型。 | |
| **Additional:** | uint64\_t num\_uint64\_t = 1364;  LogHex64{num\_uint64\_t}; | |

### [SWRD\_API\_LT\_00007] LogBin8

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00007 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00112 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogBin8 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogBin8 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint8\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示8位二进制数据类型。 | |
| **Additional:** | uint8\_t num\_uint8\_t = 15;  LogBin8{num\_uint8\_t}; | |

### [SWRD\_API\_LT\_00008] LogBin16

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00008 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00113 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogBin16 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogBin16 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint16\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示16位二进制数据类型。 | |
| **Additional:** | uint16\_t num\_uint16\_t = 315;  LogBin16{num\_uint16\_t}; | |

### [SWRD\_API\_LT\_00009] LogBin32

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00009 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00114 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogBin32 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogBin32 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint32\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示32位二进制数据类型。 | |
| **Additional:** | uint32\_t num\_uint32\_t = 1325;  LogBin32{num\_uint32\_t}; | |

### [SWRD\_API\_LT\_00010] LogBin64

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_000010 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00115 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | struct | |
| **Symbol** | LogBin64 | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct LogBin64 {…}; | |
| **Protection:** | public | |
| **Public fields:** | uint64\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 表示64位二进制数据类型。 | |
| **Additional:** | uint64\_t num\_uint64\_t = 13644;  LogBin64{num\_uint64\_t}; | |

### [SWRD\_API\_LT\_00011] Fmt

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00011 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00206、SWS\_LOG\_00225、SWS\_LOG\_00226 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | enumeration | |
| **Symbol** | Fmt | |
| **Scope:** | namespace ara::log | |
| **Underlying type:** | std::uint16\_t | |
| **Syntax:** | enum class Fmt : std::uint16\_t {...}; | |
| **Values:** | kDefault= 0 | 实现定义的格式 |
| kDec= 1 | 十进制（有符号/无符号） |
| kOct= 2 | 八进制 |
| kHex= 3 | 十六进制 |
| kBin= 4 | 二进制 |
| kDecFloat= 5 | 十进制浮点数(%f) |
| kEngFloat= 6 | 工程浮点数(%e) |
|  | kHexFloat= 7 | 十六进制浮点数(%a) |
|  | kAutoFloat= 8 | 自动“最短”浮点数(%g) |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 日志消息参数的格式说明符。  实例化 Fmt fmt;  定义了precision，使用的精度 | |

### [SWRD\_API\_LT\_00012] Format

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00012 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00207 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | struct | |
| **Symbol** | Format | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | struct Format final {...}; | |
| **Protection:** | public | |
| **Public fields:** | uint64\_t value | - |
| **Protected fields:** | None | |
| **Private fields:** | None | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 持有格式提示的类型。 | |
| **Additional:** | None | |

### [SWRD\_API\_LT\_00013] ClientState

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00013 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00098 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | enumeration | |
| **Symbol** | ClientState | |
| **Scope:** | namespace ara::log | |
| **Underlying type:** | int8\_t | |
| **Syntax:** | enum class ClientState : int8\_t {...}; | |
| **Values:** | kUnknown= -1 | - |
| kNotConnected | - |
| kConnected | - |
| **Header file:** | #include "ara/log/common.h" | |
| **Description:** | 客户端状态，表示外部客户端的连接状态。 | |

## 接口定义（API Reference）

### [SWRD\_API\_LT\_00014] CreateLogger

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00014 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00021、SWS\_LOG\_00005、SWS\_LOG\_00006、 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | CreateLogger(ara::core::StringView ctxId, ara::core::StringView ctxDescription, LogLevel ctxDefLogLevel=LogLevel::kWarn) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | Logger& CreateLogger (ara::core::StringView ctxId, ara::core::StringView ctxDescription, LogLevel ctxDefLogLevel=LogLevel::kWarn) noexcept; | |
| **Parameters (in):** | ctxId | 上下文ID。 |
| ctxDescription | 提供的上下文ID的描述。 |
| ctxDefLogLevel | 默认日志级别，如果未明确指定，则设置为警告严重性。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Logger & | 引用Logger对象的内部托管实例。所有权保留在Logging框架内 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个Logger对象，用来保存在Logging框架中注册的上下文。 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug); | |

### [SWRD\_API\_LT\_00015] HexFormat(uint8)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00015 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00022、SWS\_LOG\_00120、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(uint8\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex8 HexFormat (uint8\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex8 | 具有内置流处理句柄的LogHex8类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint8转换为十六进制值 | |
| **Additional:** | uint8\_t num\_uint8\_t = 14;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in hex format:" << HexFormat(num\_uint8\_t); | |

### [SWRD\_API\_LT\_00016] HexFormat(int8)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00016 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00023[2111M]、SWS\_LOG\_00015、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(int8\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex8 HexFormat (int8\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex8 | 具有内置流处理句柄的LogHex8类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int8转换为十六进制值 | |
| **Additional:** | int8\_t num\_int8\_t = -15;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some signed number in hex format:" << HexFormat(num\_int8\_t); | |

### [SWRD\_API\_LT\_00017] HexFormat(uint16)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00017 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00024、SWS\_LOG\_00120、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(uint16\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex16 HexFormat (uint16\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex16 | 具有内置流处理句柄的LogHex16类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint16转换为十六进制值 | |
| **Additional:** | uint16\_t num\_uint16\_t = 215;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in hex format:" << HexFormat(num\_uint16\_t); | |

### [SWRD\_API\_LT\_00018] HexFormat(int16)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00018 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00025、SWS\_LOG\_00015、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(int16\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex16 HexFormat (int16\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex16 | 具有内置流处理句柄的LogHex16类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int16转换为十六进制值 | |
| **Additional:** | int16\_t num\_int16\_t = -216;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some signed number in hex format:" << HexFormat(num\_int16\_t); | |

### [SWRD\_API\_LT\_00019] HexFormat(uint32)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00019 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00026、SWS\_LOG\_00120、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(uint32\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex32 HexFormat (uint32\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex32 | 具有内置流处理句柄的LogHex32类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint32转换为十六进制值 | |
| **Additional:** | uint32\_t num\_uint32\_t = 235;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in hex format:" << HexFormat(num\_uint32\_t); | |

### [SWRD\_API\_LT\_00020] HexFormat(int32)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00020 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00027、SWS\_LOG\_00015、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(int32\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex32 HexFormat (int32\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex32 | 具有内置流处理句柄的LogHex32类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int32转换为十六进制值 | |
| **Additional:** | int32\_t num\_int32\_t = -230;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some signed number in hex format:" << HexFormat(num\_int32\_t); | |

### [SWRD\_API\_LT\_00021] HexFormat(uint64)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00021 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00028、SWS\_LOG\_00120、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(uint64\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex64 HexFormat (uint64\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex64 | 具有内置流处理句柄的LogHex64类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint64转换为十六进制值 | |
| **Additional:** | uint64\_t num\_uint64\_t = 3654;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in hex format:" << HexFormat(num\_uint64\_t); | |

### [SWRD\_API\_LT\_00022] HexFormat(int64)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00022 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00029、SWS\_LOG\_00015、SWS\_LOG\_00016 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | HexFormat(int64\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogHex64 HexFormat (int64\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为十六进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogHex64 | 具有内置流处理句柄的LogHex64类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int64转换为十六进制值 | |
| **Additional:** | int64\_t num\_int64\_t = -1542;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some signed number in hex format:" << HexFormat(num\_int64\_t); | |

### [SWRD\_API\_LT\_00023] BinFormat(uint8)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00023 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00030[2111M]、SWS\_LOG\_00120、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(uint8\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin8 BinFormat (uint8\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin8 | 具有内置流处理句柄的LogBin8类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint8转换为二进制值 | |
| **Additional:** | uint8\_t num\_uint8\_t = 14;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_uint8\_t); | |

### [SWRD\_API\_LT\_00024] BinFormat(int8)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00024 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00031、SWS\_LOG\_00015、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(int8\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin8 BinFormat (int8\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin8 | 具有内置流处理句柄的LogBin8类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int8转换为二进制值 | |
| **Additional:** | int8\_t num\_int8\_t = -15;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_int8\_t); | |

### [SWRD\_API\_LT\_00025] BinFormat(uint16)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00025 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00032、SWS\_LOG\_00120、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(uint16\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin16 BinFormat (uint16\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin16 | 具有内置流处理句柄的LogBin16类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint16转换为二进制值 | |
| **Additional:** | uint16\_t num\_uint16\_t = 215;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_uint16\_t); | |

### [SWRD\_API\_LT\_00026] BinFormat(int16)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00026 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00033、SWS\_LOG\_00015、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | BinFormat(int16\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin16 BinFormat (int16\_t value) noexcept; | |
| **Parameters (in):** | value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin16 | 具有内置流处理句柄的LogBin16类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int16转换为二进制值 | |
| **Additional:** | int16\_t num\_int16\_t = -215;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_int16\_t); | |

### [SWRD\_API\_LT\_00027] BinFormat(uint32)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00027 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00034、SWS\_LOG\_00120、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(uint32\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin32 BinFormat (uint32\_t value) noexcept; | |
| **Parameters (in):** | Value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin32 | 具有内置流处理句柄的LogBin32类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint32转换为二进制值 | |
| **Additional:** | uint32\_t num\_uint32\_t = 322;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_uint32\_t); | |

### [SWRD\_API\_LT\_00028] BinFormat(int32)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00028 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00035、SWS\_LOG\_00015、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(int32\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin32 BinFormat (int32\_t value) noexcept; | |
| **Parameters (in):** | Value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin32 | 具有内置流处理句柄的LogBin32类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int32转换为二进制值 | |
| **Additional:** | int32\_t num\_int32\_t = -325;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_int32\_t); | |

### [SWRD\_API\_LT\_00029] BinFormat(uint64)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00029 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00036、SWS\_LOG\_00120、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(uint64\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin64 BinFormat (uint64\_t value) noexcept; | |
| **Parameters (in):** | Value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin64 | 具有内置流处理句柄的LogBin64类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将uint32转换二进制值 | |
| **Additional:** | uint64\_t num\_uint64\_t = 1322;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_uint64\_t); | |

### [SWRD\_API\_LT\_00030] BinFormat(int64)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00030 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00037、SWS\_LOG\_00015、SWS\_LOG\_00017 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | BinFormat(int64\_t value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr LogBin64 BinFormat (int64\_t value) noexcept; | |
| **Parameters (in):** | Value | 要转换为二进制数字系统的十进制数。 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogBin64 | 具有内置流处理句柄的LogBin64类型。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 将int64转换为二进制值 | |
| **Additional:** | int64\_t num\_int64\_t = -1322;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << " some unsigned number in bin format:" << BinFormat(num\_int64\_t); | |

### [SWRD\_API\_LT\_00031] Wrapper object creator

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00031 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00201 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | Arg(T &&arg, const char \*name=nullptr, const char \*unit=nullptr) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | template <typename T>  Argument<T> Arg (T &&arg, const char \*name=nullptr, const char\*unit=nullptr) noexcept; | |
| **Parameters (in):** | arg | 参数负载对象 |
| name | “arg”可选的“名称”属性 |
| uint | “arg”可选的“单位”属性 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Argument<T> | 包含提供的参数的封装对象。 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 为给定的参数创建封装对象。  如果满足以下任何条件，则调用该函数的格式将不正确：T不是算术类型，不是bool，并且不能转换为ara::core::StringView，也不能转换为 ara::core::Span <const ara::core::Byte>，T可转换为ara::core::StringView或可转换为ara :: core :: Span <const ara :: core :: Byte>或bool，并且unit不是nullptr | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << Arg("text", "identifier"); | |

### [SWRD\_API\_LT\_00032] Logger of an argument with attributes

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00032 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00203 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<(const Argument< T > &arg) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | template <typename T> LogStream& operator<< (const Argument< T > &arg) noexcept; | |
| **Parameters (in):** | T | 参数负载类型 |
| Arg | 参数负载对象 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 记录带有属性的参数。当输出到控制台时，该值及其所有属性将作为一个参数显示。其他模式下，也作为一个参数显示。 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << Arg("text", "identifier"); | |

### [SWRD\_API\_LT\_00033] Class LogStream

|  |  |
| --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00033 |
| **TYPE** | Valid |
| **Priority** | H |
| **Upstream ID** | - |
| **CR** | - |
| **Consistency** | Yes |
| **Change Type** | 不变 |
| ***Kind:*** | Class |
| **Symbol:** | LogStream |
| **Scope:** | namespace ara::log |
| **Syntax:** | LogStream() = delete;  LogStream(LogLevel logLevel, Logger& logger) noexcept;  LogStream(const LogStream&) = delete;  LogStream& operator=(const LogStream&) = delete;  LogStream(LogStream&&);  LogStream& operator=(LogStream&&) = delete; |
| **Header file:** | #include "ara/log/log\_stream.h" |
| **Description:** | ara::log::LogStream表示日志消息，允许流操作符用于附加数据。  单条日志最大记录长度是1390字节。 |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  LogStream LOG\_STREAMER\_UUU2\_ERROR{MY\_LOGGER.LogError()};  LOG\_STREAMER\_UUU2\_ERROR << " message......";  LOG\_STREAMER\_UUU2\_ERROR << " message2......";  LOG\_STREAMER\_UUU2\_ERROR.Flush(); |

### [SWRD\_API\_LT\_00034] LogStream::Flush

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00034 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00039 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | Flush() | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | void Flush() noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | None |  |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 发送当前日志缓冲区并发起一个新的消息流。 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  LogStream LOG\_STREAMER\_UUU2\_ERROR{MY\_LOGGER.LogError()};  LOG\_STREAMER\_UUU2\_ERROR << " message1......";  LOG\_STREAMER\_UUU2\_ERROR << " message2......";  LOG\_STREAMER\_UUU2\_ERROR.Flush();  LOG\_STREAMER\_UUU2\_ERROR << " message3......";  LOG\_STREAMER\_UUU2\_ERROR << " message4......";  LOG\_STREAMER\_UUU2\_ERROR.Flush(); | |

### [SWRD\_API\_LT\_00035] operator<< (uint8\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00035 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00041 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(uint8\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (uint8\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | uint8\_t num\_uint8\_t = 14;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << num\_uint8\_t;  LogStream LOG\_STREAMER\_UUU2\_ERROR{MY\_LOGGER.LogError()}; | |

### [SWRD\_API\_LT\_00036] operator<<(bool value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00036 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00040 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<(bool value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (bool value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | bool boo = false;  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << boo;  LogStream LOG\_STREAMER\_UUU2\_ERROR{MY\_LOGGER.LogError()};  LOG\_STREAMER\_UUU2\_ERROR << boo; | |

### [SWRD\_API\_LT\_00037] operator<< (uint16\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00037 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00042 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(uint16\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (uint16\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Return value:** | LogStream& | \*this |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00038] operator<< (uint32\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00038 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00043、User\_Defined\_LOG\_00037 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(uint32\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (uint32\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout)** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00039] operator<< (uint64\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00039 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00044 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(uint64\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (uint64\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00040] operator<< (int8\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00040 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00045 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(int8\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (int8\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | Reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00041] operator<< (int16\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00041 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00046 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(int16\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (int16\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00042] operator<< (int32\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00042 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00047 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(int32\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (int32\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | Reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00043] operator<< (int64\_t value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00043 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00048 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(int64\_t value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (int64\_t value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00044] operator<<(float value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00044 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00049 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(float value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<<(float value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00045] operator<<(double value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00045 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00050 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<(double value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<<(double value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00046] operator<<(const LogHex8 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00046 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00053 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<(const LogHex8 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogHex8 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00047] operator<<(const LogHex16 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00047 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00054 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogHex16 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogHex16 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00048] operator<<(const LogHex32 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00048 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00055 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogHex32 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogHex32 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00049] operator<<(const LogHex64 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00049 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00056 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogHex64 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogHex64 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00050] operator<<(const LogBin8 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00050 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00057 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogBin8 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogBin8 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00051] operator<<(const LogBin16 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00051 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00058 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogBin16 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogBin16 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00052] operator<<(const LogBin32 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00051 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00059 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogBin32 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogBin32 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00053] operator<<(const LogBin64 &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00053 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00060 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const LogBin64 &value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const LogBin64 &value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00054] operator<<(const ara::core::StringView value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00054 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00062 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const ara::core::StringView value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const ara::core::StringView value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | std::string aa = "string";  ara::core::StringView stwi{aa.c\_str(), aa.size()};  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug); | |

### [SWRD\_API\_LT\_00055] operator<<(const char \*const value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00055 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00063 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(const char \*const value) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (const char \*const value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | const char \*cca = "cca";  char \*const ccb = "ccb";  const char \*const ccc = "ccc";  Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << cca;  MY\_LOGGER.LogInfo() << ccb;  MY\_LOGGER.LogInfo() << ccc;  LogStream LOG\_STREAMER\_UUU2\_ERROR{MY\_LOGGER.LogError()};  LOG\_STREAMER\_UUU2\_ERROR << cca;  LOG\_STREAMER\_UUU2\_ERROR << ccb;  LOG\_STREAMER\_UUU2\_ERROR << ccc; | |

### [SWRD\_API\_LT\_00056] operator<<(LogStream &out, LogLevel value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00056 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00063 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(LogStream &out, LogLevel value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | LogStream& operator<< (LogStream &out, LogLevel value) noexcept; | |
| **Parameters (in):** | Value | 要附加到内部消息缓冲区的值 |
| **Parameters (inout):** | Out | 附加上值的LogStream对象 |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | - |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | reentrant | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << LogLevel::kDebug; | |

### [SWRD\_API\_LT\_00057] operator<<(LogStream &out, const core::ErrorCode &ec)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00057 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00124 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<(LogStream &out, const core::ErrorCode &ec) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | LogStream& operator<< (LogStream &out, LogLevel value) noexcept; | |
| **Parameters (in):** | Ec | 要记录的ErrorCode实例 |
| **Parameters (inout):** | Out | 附加上值的LogStream对象 |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | - |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00058] operator<<(LogStream &out, const std::chrono::duration< Rep, Period > &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00058 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00125 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<(LogStream &out, const std::chrono::duration< Rep, Period > &value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | template <typename Rep, typename Period>  LogStream& operator<< (LogStream &out, const std::chrono::duration<  Rep, Period > &value) noexcept; | |
| **Template param:** | Rep | 表示此持续时间内的tick数的算术类型 |
| Period | 一种std::ratio类型，表示时钟的tick周期，以秒为单位 |
| **Parameters (in):** | Value | 要记录的duration实例 |
| **Parameters (inout):** | Out | 附加上值的LogStream对象 |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | - |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00059] operator<<(LogStream &out, const ara::core::InstanceSpecifier &value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00059 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00126 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<( LogStream &out, const ara::core::InstanceSpecifier &value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | LogStream& operator<< (LogStream &out, const ara::core::InstanceSpecifier &value) noexcept; | |
| **Parameters (in):** | Value | 要记录的duration实例 |
| **Parameters (inout):** | Out | 附加上值的InstanceSpecifier对象 |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | - |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00060] operator<<(LogStream &out, const void \*const value)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00060 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00127 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | operator<<(LogStream &out, const void \*const value) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | LogStream & operator<< (LogStream &out, const void \*const value)  noexcept; | |
| **Parameters (in):** | Value | 记录 |
| **Parameters (inout):** | Out | 要向其中添加值的 LogStream 对象 |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | - |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将指针地址写入消息，格式为十六进制 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00061] operator<<( core::Span< const core::Byte > data)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00061 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00128 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | function | |
| **Symbol:** | operator<<( core::Span< const core::Byte > data) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& operator<< (core::Span< const core::Byte > data) noexcept; | |
| **Parameters (in):** | Data | a Span<const Byte>覆盖要记录的范围 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将给定的值追加到内部消息缓冲区。 | |
| **Additional:** | 参见5.3.23，只是更改下数据类型 | |

### [SWRD\_API\_LT\_00062] WithLocation(core::StringView file, int line)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00062 | |
| **TYPE** | Valid | |
| **Priority** | M | |
| **Upstream ID** | SWS\_LOG\_00129 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | function | |
| **Symbol:** | WithLocation(core::StringView file, int line) | |
| **Scope:** | class ara::log::LogStream | |
| **Syntax:** | LogStream& WithLocation (core::StringView file, int line) noexcept; | |
| **Parameters (in):** | File | 源文件标识符 |
| Line | 源文件行号 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream& | \*this |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/log\_stream.h" | |
| **Description:** | 将源文件位置添加到消息中。  如果已经调用了另一个向当前消息添加内容的成员函数，则此函数将不起作用。 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo().WithLocation(\_\_FILE\_\_, \_\_LINE\_\_) << …; | |

### [SWRD\_API\_LT\_00063] Logger::LogFatal

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00063 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00064 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | LogFatal() | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream LogFatal () const noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | Fatal级别LogStream对象 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个LogStream对象。  返回的对象将通过插入流操作符“<<”接受参数 | |
| **Notes:** | 在正常使用情况下，创建的LogStream的对象生存期在一个语句内（在最后一个传递的参数之后以;结尾）。 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogFatal() << 123; | |

### [SWRD\_API\_LT\_00064] Logger::LogError

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00064 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00065 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | LogError() | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream LogError() const noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | Error级别LogStream对象 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个LogStream对象。  返回的对象将通过插入流操作符“<<”接受参数 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogError() << 123; | |

### [SWRD\_API\_LT\_00065] Logger::LogWarn

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00065 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00066 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | LogWarn() | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream LogWarn() const noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | Warn级别LogStream对象 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个LogStream对象。  返回的对象将通过插入流操作符“<<”接受参数 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogWarn() << 123; | |

### [SWRD\_API\_LT\_00066] Logger::LogInfo

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00066 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00067 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | LogInfo() | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream LogInfo () const noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | Info级别LogStream对象 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个LogStream对象。  返回的对象将通过插入流操作符“<<”接受参数 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogInfo() << 123; | |

### [SWRD\_API\_LT\_00067] Logger::LogDebug

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00067 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00068 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | LogDebug() | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream LogDebug () const noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | Debug级别LogStream对象 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个LogStream对象。  返回的对象将通过插入流操作符“<<”接受参数 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogDebug() << 123; | |

### [SWRD\_API\_LT\_00068] Logger::LogVerbose

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00068 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00069 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | LogVerbose() | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream LogVerbose () const noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | Verbose级别LogStream对象 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 创建一个LogStream对象。  返回的对象将通过插入流操作符“<<”接受参数 | |
| **Additional:** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kDebug);  MY\_LOGGER.LogVerbose() << 123; | |

### [SWRD\_API\_LT\_00069] Logger::IsEnabled

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00069 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00070、SWS\_LOG\_00070 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 不变 | |
| ***Kind:*** | Function | |
| **Symbol:** | IsEnabled(LogLevel logLevel) | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | bool IsEnabled (LogLevel logLevel) const noexcept; | |
| **Parameters (in):** | LogLevel | 需要检查的日志级别 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Bool | 如果相应logLevel的日志会被输出的话则返回true，否则返回false |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 查看当前配置的日志报告级别。 | |
| **Additional** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kVerbose);  MY\_LOGGER.LogInfo() << "kVerbose enabled:" << MY\_LOGGER.IsEnabled(LogLevel::kVerbose) | |

### [SWRD\_API\_LT\_00070] Logger::WithLevel

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00070 | |
| **TYPE** | Valid | |
| **Priority** | H | |
| **Upstream ID** | SWS\_LOG\_00131 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | function | |
| **Symbol:** | WithLevel(LogLevel logLevel) | |
| **Scope:** | class ara::log::Logger | |
| **Syntax:** | LogStream WithLevel (LogLevel logLevel) const noexcept; | |
| **Parameters (in):** | logLevel | 日志级别 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | 具有给定日志级别的新LogStream实例 |
| **Exception Safety:** | noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 可以编写以编程方式确定日志级别的日志消息 | |
| **Additional** | Logger& MY\_LOGGER = CreateLogger("MYLG", "my default context 1",LogLevel::kVerbose);  MY\_LOGGER. WithLevel(LogLevel::kInfo) << " hello word! " ; | |

### [SWRD\_API\_LT\_00071] RegisterConnectionStateHandler(std::function< void(ClientState)> callback)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00071 | |
| **TYPE** | Valid | |
| **Priority** | N/A | |
| **Upstream ID** | SWS\_LOG\_00205 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | RegisterConnectionStateHandler(std::function< void(ClientState)> callback) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | void RegisterConnectionStateHandler (std::function< void(ClientState)> callback) noexcept; | |
| **Parameters (in):** | Callback | 连接状态更改时调用的回调 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | LogStream | 具有给定日志级别的新LogStream实例 |
| **Exception Safety:** | Noexcept | |
| **Thread Safety:** | - | |
| **Header file:** | #include "ara/log/logger.h" | |
| **Description:** | 注册一个回调，以便在连接状态更改时调用 | |
| **Additional** | None | |

### [SWRD\_API\_LT\_00072] Dflt()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00072 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00208 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Dflt() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Dflt () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kDefault 格式提示创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00073] Hex()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00073 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00209 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Hex() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Hex () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kHex 格式提示创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00074] Hex(std::uint16\_t precision)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00074 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00210 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Hex(std::uint16\_t precision) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Hex (std::uint16\_t precision) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kHex 格式提示和给定精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00075] Dec()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00075 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00211 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Dec() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Dec () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kDec 格式提示和默认精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00076] Dec(std::uint16\_t precision)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00076 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00212 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Dec(std::uint16\_t precision) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Dec (std::uint16\_t precision) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kDec 格式化提示和给定精度创建一个 Format 实例 | |

### [SWRD\_API\_LT\_00077] Oct()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00077 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00213 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Oct() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Oct () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kOct 格式提示和默认精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00078] Oct(std::uint16\_t precision)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00078 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00214 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Oct(std::uint16\_t precision) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Oct (std::uint16\_t precision) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kOct 格式提示和给定精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00079] Bin()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00079 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00215 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Bin() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Bin () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kBin 格式化提示和默认精度创建一个 Format 实例 | |

### [SWRD\_API\_LT\_00080] Bin(std::uint16\_t precision)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00080 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00216 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | Bin(std::uint16\_t precision) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format Dflt () noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kBin 格式提示和给定精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00081] DecFloat(std::uint16\_t precision=6)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00081 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00217 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | DecFloat(std::uint16\_t precision=6) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format DecFloat (std::uint16\_t precision=6) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kDecFloat 格式化提示和给定精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00082] DecFloatMax()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00082 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00218 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | DecFloatMax() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format DecFloatMax () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kDecFloat 格式提示和足以保证完整往返安全的精度创建一个 Format 实例 | |

### [SWRD\_API\_LT\_00083] EngFloat(std::uint16\_t precision=6)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00083 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00219 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | EngFloat(std::uint16\_t precision=6) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format EngFloat (std::uint16\_t precision=6) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kEngFloat 格式化提示和给定精度创建一个 Format 实例 | |

### [SWRD\_API\_LT\_00084] EngFloatMax()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00084 | |
| **TYPE** | Valid | |
| **Priority** | ; | |
| **Upstream ID** | SWS\_LOG\_00220 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | EngFloatMax() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format EngFloatMax () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kEngFloat 格式化提示和足以保证完整往返安全的精度创建一个 Format 实例 | |

### [SWRD\_API\_LT\_00085] HexFloat(std::uint16\_t precision)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00085 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00221 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | HexFloat(std::uint16\_t precision) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format HexFloat (std::uint16\_t precision) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kHexFloat 格式提示和给定精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00086] HexFloatMax()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00086 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00222 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | HexFloatMax() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format HexFloatMax () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kHexFloat 格式提示和足以保证完整往返安全的精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00087] AutoFloat(std::uint16\_t precision=6)

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00087 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00223 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | AutoFloat(std::uint16\_t precision=6) | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format AutoFloat (std::uint16\_t precision=6) noexcept; | |
| **Parameters (in):** | Precision | 使用精度 |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kAutoFloat 格式化提示和给定精度创建一个 Format 实例。 | |

### [SWRD\_API\_LT\_00088] AutoFloatMax()

|  |  |  |
| --- | --- | --- |
| **SWRD\_ID*:*** | SWRD\_API\_LT\_00088 | |
| **TYPE** | Valid | |
| **Priority** | L | |
| **Upstream ID** | SWS\_LOG\_00224 | |
| **CR** | - | |
| **Consistency** | Yes | |
| **Change Type** | 新增 | |
| ***Kind:*** | Function | |
| **Symbol:** | AutoFloatMax() | |
| **Scope:** | namespace ara::log | |
| **Syntax:** | constexpr Format AutoFloatMax () noexcept; | |
| **Parameters (in):** | None |  |
| **Parameters (inout):** | None |  |
| **Parameters (out):** | None |  |
| **Return value:** | Format | 一个Format类型实例 |
| **Description:** | 使用 Fmt::kAutoFloat 格式化提示和足以保证完整往返安全的精度创建一个 Format 实例 | |

附录A- 信息定义

|  |  |  |
| --- | --- | --- |
| 类别 | 结构 | 备注 |
| 需求 | SWRD\_{需求类型}\_{功能简称}\_流水号  功能简称：参见下面功能简称列表  需求类型：功能需求为空，非功能需求为NF,接口为API  流水号：从00001开始的5位自然数 | *例：*  *SWRD\_Nvm\_00001*  *SWRD\_NF\_Nvm\_00001*  *SWRD\_API\_Nvm\_00001* |

|  |  |
| --- | --- |
| 功能简称列表（aCore） | 说明 |
| DM\_DEM | Diagnostics management模块的诊断事件管理 |
| DM\_DCM | Diagnostics management模块的诊断通信管理 |
| DM\_DCM\_DOIP | Diagnostics management模块的DO/IP相关功能 |
| CoreTypes | 核心数据类型 |
| CM\_SOMEIP | Communication management模块的SOME/IP相关功能 |
| CM\_DDS | Communication management模块的DDS相关功能 |
| CM\_CommunicationGroup | Communication management模块的通信组相关功能 |
| CM\_SHM | Communication management模块的共享内存相关功能 |
| CM\_IPC | Communication management模块的IPC相关功能 |
| CM\_Raw | Communication management模块的raw data streaming相关功能 |
| CM\_TLS | Communication management模块的TLS相关功能 |
| CM\_S2S | Communication management模块的S2S相关功能 |
| CM\_E2E | Communication management模块的E2E相关功能 |
| UCM\_Master | Update and config management模块的主站相关功能 |
| UCM\_Server | Update and config management模块的从站相关功能 |
| LT | Log and trace模块相关功能 |
| PHM | Platform health management模块相关功能 |
| Per | Persistency模块相关功能 |
| SM | State management模块相关功能 |
| Crypto | Cryptography模块相关功能 |
| EM | Execution mangement模块相关功能 |
| NM | Network management模块相关功能 |
| TS | Time synchronization模块相关功能 |

说明：根据项目情况可自己定义，增加功能简称

|  |  |  |
| --- | --- | --- |
| 安全等级(ASIL) | 解释说明 | 备注 |
| ASIL A | 根据S – Severity(严重度)  E – Exposure（暴露度） C – Controllability（可控性） 排定功能安全等级。详细理解可以参考26262标准文件。 | *如果有关于ASIL等级的特殊解释说明，请记录在此* |
| ASIL B |  |
| ASIL C |  |
| ASIL D |  |
| QM(A) | 从ASIL A到ASIL D 中拆分出来，拆分的标准，参考功能安全体系文件《功能安全需求分解指南\_FS.pdf》 |  |
| QM(B) |  |
| QM(C) |  |
| QM(D) |  |
| ASIL A(A) |  |
| ASIL A(B) |  |
| ASIL A(C) |  |
| ASIL A(D) |  |
| ASIL B(B) |  |
| ASIL B(C) |  |
| ASIL B(D) |  |
| ASIL C(C) |  |
| ASIL C(D） |  |
| ASIL D(D) |  |

|  |  |  |
| --- | --- | --- |
| 优先级（Priority） | 解释说明 | 备注 |
| H | 高优先级 | *例：被依赖的需求优先级设置为H级别* |
| M | 中优先级 | *例：* |
| L | 低优先级 | *例：其余功能均设置为L级别* |

|  |  |  |
| --- | --- | --- |
| 状态  （Status） | 状态说明 | 备注 |
| Draft | 草稿 | *例：表示新建* |
| In Review | 评审中 | *例：表示处于评审中* |
| Approved | 批准 | *例：表示通过评审* |
| Released | 发布 | *例：表示通过客户确认* |
| Modified | 修改 | *例：表示正在检讨修改中* |

|  |  |  |
| --- | --- | --- |
| 类型  （Type） | 状态说明 | 备注 |
| Valid | 有效 | *例：表示需要对应* |
| InValid | 不适用 | *例：表示不做对应* |
| TBD | 检讨中 | *例：表示正在检讨中* |

说明：根据项目情况可自己定义，但需要明确

|  |  |  |
| --- | --- | --- |
| 变更类型  (Change Type) | 解释说明 | 备注 |
| 新增 | 相对已建立的第一版需求基线（含Base项目的需求基线），如果是新增的需求，选择此项 | 如果有关于每个变更类型的特殊解释说明，请记录在此 |
| 修改 | 相对已建立的第一版需求基线（含Base项目的需求基线），发生了修改的需求 |  |
| 不变 | 相对已建立的base项目的需求基线，复用了base项目的需求，填此类型，如没有Base项目需求基线，不应填此类型。 |  |
| 删除 | 相对已建立的第一版需求基线（含Base项目的需求基线），如果是删除的需求，选择此项。 |  |

说明：根据项目情况可自己定义，但需要明确

附录B- 配置信息

|  |  |  |  |
| --- | --- | --- | --- |
| 配置信息 | 说明 | 范围 | 备注 |
| *API configuration class* |  | *1、2、3* |  |
|  |  |  |  |
|  |  |  |  |