PD ISO/IEC TS 19568:2017

BSI Standards Publication

Programming languages — C++ extensions for library fundamentals

bsi.

National foreword

This Published Document is the UK implementation of ISO/IEC TS 19568:2017. It supersedes PD ISO/IEC TS 19568:2015 which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee IST/5, Programming languages, their environments and system software interfaces.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Published by BSI Standards Limited 2017

ISBN 978 0 580 93295 3
ICS 35.060

Compliance with a British Standard cannot confer immunity from legal obligations.

This Published Document was published under the authority of the Standards Policy and Strategy Committee on 30 April 2017.

Amendments/corrigenda issued since publication

Date	Text affected

TECHNICAL SPECIFICATION

ISO/IEC TS 19568

Second edition
2017-03

Programming languages — C++ extensions for library fundamentals

Langages de programmation — Extensions C++ pour la bibliothèque fondamentaux

Reference number
ISO/IEC TS 19568:2017(E)

Contents

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement. For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, Information Technology, Subcommittee SC 22, Programming languages, their environments and system software interfaces. This edition of ISO/IEC 19568:2017 cancels and replaces the edition ISO/IEC 19568:2015, which has been technically revised and includes the following changes:

- Addition of the `sample` algorithm.
- Addition of new random-number generation facilities, and algorithms which use them.
- Addition of algorithms for uniform container erasure.
- Addition of function template `not_fn`.
- Addition of logical operator type traits `conjunction`, `disjunction`, and `negation`.
- Addition of templates to support the "detection idiom".
- Addition of the `propagate_const` class template.
- Addition of the `observer_ptr` class template.
- Addition of the `make_array` and `to_array` function templates.
- Addition of the `ostream_joiner` class template.
- Addition of the `gcd` and `lcm` algorithms.
- Addition of the `source_location` struct.
- Changes to the return types of search algorithms.
- Moving all libraries to the inline namespace `fundamentals_v2`.
- Miscellaneous defect resolutions.

1 General [general]

1.1 Scope [general.scope]

1 This technical specification describes extensions to the C++ Standard Library (1.2). These extensions are classes and functions that are likely to be used widely within a program and/or on the interface boundaries between libraries written by different organizations.

2 This technical specification is non-normative. Some of the library components in this technical specification may be considered for standardization in a future version of C++, but they are not currently part of any C++ standard. Some of the components in this technical specification may never be standardized, and others may be standardized in a substantially changed form.

3 The goal of this technical specification is to build more widespread existing practice for an expanded C++ standard library. It gives advice on extensions to those vendors who wish to provide them.

1.2 Normative references [general.references]

1 The following referenced document is indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

— ISO/IEC 14882:2014, *Programming Languages — C++*

2 ISO/IEC 14882:2014 is herein called the *C++ Standard*. References to clauses within the C++ Standard are written as "C++14 §3.2". The library described in ISO/IEC 14882:2014 clauses 17–30 is herein called the *C++ Standard Library*.

3 Unless otherwise specified, the whole of the C++ Standard's Library introduction (C++14 §17) is included into this Technical Specification by reference.

1.3 Namespaces, headers, and modifications to standard classes [general.namespaces]

1 Since the extensions described in this technical specification are experimental and not part of the C++ standard library, they should not be declared directly within namespace `std`. Unless otherwise specified, all components described in this technical specification either:

— modify an existing interface in the C++ Standard Library in-place,
— are declared in a namespace whose name appends `::experimental::fundamentals_v2` to a namespace defined in the C++ Standard Library, such as `std` or `std::chrono`, or
— are declared in a subnamespace of a namespace described in the previous bullet, whose name is not the same as an existing subnamespace of namespace `std`.

[*Example:* This TS does not define `std::experimental::fundamentals_v2::chrono` because the C++ Standard Library defines `std::chrono`. This TS does not define `std::pmr::experimental::fundamentals_v2` because the C++ Standard Library does not define `std::pmr`. — *end example*]

2 Each header described in this technical specification shall import the contents of `std::experimental::fundamentals_v2` into `std::experimental` as if by

```
namespace std {
  namespace experimental {
    inline namespace fundamentals_v2 {}
```