TECHNICAL REPORT

ISO/IEC TR 19768

First edition
2007-11-15

Information technology — Programming languages — Technical Report on C++ Library Extensions

Technologies de l'information — Langages de programmation — Rapport technique sur l'extensions des librairies C++

Reference number
ISO/IEC TR 19768:2007(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

In exceptional circumstances, the joint technical committee may propose the publication of a Technical Report of one of the following types:

— type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;

— type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;

— type 3, when the joint technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC TR 19768, which is a Technical Report of type 2, was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces.*

List of Tables

1 Scope [tr.intro]

1 This Technical Report describes extensions to the *C++ standard library* that is described in the International Standard for the C++ programming language [14].

2 This Technical Report is non-normative. Some of the library components in this Technical Report 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 Report may never be standardized, and others may be standardized in a substantially changed form.

3 The goal of this Technical Report 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.

1.1 Relation to C++ Standard Library Introduction [tr.description]

1 Unless otherwise specified, the whole of the ISO C++ Standard Library introduction [lib.library] is included in this Technical Report by reference.

1.2 Categories of extensions [tr.intro.ext]

1 This Technical Report describes four general categories of library extensions:

1. New requirement tables, such as the regular expression traits requirements in 7.2. These are not directly expressed as software; they specify the circumstances under which user-written components will interoperate with the components described in this Technical Report.

2. New library components (types and functions) that are declared in entirely new headers, such as the class templates in the <unordered_set> header (6.3.4.1).

3. New library components declared as additions to existing standard headers, such as the mathematical special functions added to the headers <cmath> and <math.h> in 5.2.1 and 5.2.2

4. Additions to standard library components, such as the extensions to class std::pair in 6.1.4.

2 New headers are distinguished from extensions to existing headers by the title of the *synopsis* clause. In the first case, the title is of the form “Header <foo> synopsis”, and the synopsis includes all namespace scope declarations contained in the header. In the second case, the title is of the form “Additions to header <foo> synopsis” and the synopsis includes only the extensions, *i.e.* those namespace scope declarations that are not present in the C++ standard [14].

1.3 Namespaces and headers [tr.intro.namespaces]

1 Since the extensions described in this Technical Report are not part of the C++ standard library, they should not be declared directly within namespace std. Unless otherwise specifed, all components described in this Technical Report are declared in namespace std::tr1. [*Note*: Some components are declared in subnamespaces of namespace std::tr1. —*end note*]

2 Unless otherwise specified, references to other entities described in this Technical Report are assumed to be qualified with std::tr1::, and references to entities described in the International Standard are assumed to be qualified with std::.

3 Even when an extension is specified as additions to standard headers (the third category in 1.2), vendors should not simply add declarations to standard headers in a way that would be visible to users by default. [*Note*: That would fail to be standard conforming, because the new names, even within a namespace, could conflict with user macros. —*end note*] Users should be required to take explicit action to have access to library extensions.

4 It is recommended either that additional declarations in standard headers be protected with a macro that is not defined by default, or else that all extended headers, including both new headers and parallel versions of standard headers with non-standard declarations, be placed in a separate directory that is not part of the default search path.

2 General Utilities [tr.util]

1 This clause describes basic components used to implement other library facilities. They may also be used by C++ programs.

2 The following subclauses describe reference wrappers and smart pointers, as summarized in Table 1.

Table 1: Utilities library summary

Subclause	Header(s)
2.1 Reference wrapper	`<functional>`
2.2 Smart pointers	`<memory>`

2.1 Reference wrappers [tr.util.refwrap]

2.1.1 Additions to header `<functional>` synopsis [tr.util.refwrp.synopsis]


```
namespace std {
namespace tr1 {
  template <class T> class reference_wrapper;

  template <class T> reference_wrapper<T> ref(T&);
  template <class T> reference_wrapper<const T> cref(const T&);

  template <class T> reference_wrapper<T> ref(reference_wrapper<T>);
  template <class T> reference_wrapper<const T> cref(reference_wrapper<T>);
} // namespace tr1
} // namespace std
```

2.1.2 Class template `reference_wrapper` [tr.util.refwrp.refwrp]

```
template <class T> class reference_wrapper
  : public unary_function<T1, R>          // see below
  : public binary_function<T1, T2, R>     // see below
{
public :
  // types
  typedef T type;
  typedef see below result_type; // Not always defined
```