International Standard

ISO/IEC/IEEE 9945

Information technology — Portable Operating System Interface (POSIX™) Base Specifications, Issue 8

Technologies de l'information — Spécifications de base de l'interface pour la portabilité des systèmes (POSIX™), Issue 8

Second edition
2026-03

Reference number
ISO/IEC/IEEE 9945:2026(en)

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.

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 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 or www.iec.ch/members_experts/refdocs).

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of 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 www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

ISO/IEC/IEEE 9945 was prepared by the Microprocessor Committee of the IEEE Computer Society (as IEEE Std 1003.1-2024) and The Open Group (as The Open Group Technical Standard Base Specifications, Issue 8) and drafted in accordance with its editorial rules. It was adopted, under the “fast-track procedure” defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE, by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces*.

This second edition cancels and replaces the first edition (ISO/IEC/IEEE 9945:2009), which has been technically revised. It also incorporates the Technical Corrigenda ISO/IEC/IEEE 9945:2009/Cor. 1:2013 and ISO/IEC/IEEE 9945:2009/Cor. 2:2017.

The main changes are as follows:

— Change history is described in the Rationale (Informative) volume of POSIX.1-2024 and in the CHANGE HISTORY section of reference pages.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

IEEE Std 1003.1™-2024
(Revision of IEEE Std 1003.1-2017)

The Open Group Standard, Base Specifications, Issue 8

IEEE Standard for Information Technology—Portable Operating System Interface (POSIX™)

Base Specifications, Issue 8

Developed by the

Microprocessor Committee
of the
IEEE Computer Society

and

The Open Group

Approved 20 May 2024

IEEE SA Standards Board

IEEE Std 1003.1™-2024 (Revision of IEEE Std 1003.1-2017)
IEEE Standard for Information Technology—Portable Operating System Interface (POSIX®)
The Open Group Standard, Base Specifications, Issue 8

Abstract: POSIX.1-2024 is simultaneously IEEE Std 1003.1™-2024 and The Open Group Standard Base Specifications, Issue 8.

POSIX.1-2024 defines a standard operating system interface and environment, including a command interpreter (or "shell"), and common utility programs to support applications portability at the source code level. POSIX.1-2024 is intended to be used by both application developers and system implementors and comprises four major components (each in an associated volume):

- General terms, concepts, and interfaces common to all volumes of this standard, including utility conventions and C-language header definitions, are included in the Base Definitions volume.
- Definitions for system service functions and subroutines, language-specific system services for the C programming language, function issues, including portability, error handling, and error recovery, are included in the System Interfaces volume.
- Definitions for a standard source code-level interface to command interpretation services (a "shell") and common utility programs for application programs are included in the Shell and Utilities volume.
- Extended rationale that did not fit well into the rest of the document structure, which contains historical information concerning the contents of POSIX.1-2024 and why features were included or discarded by the standard developers, is included in the Rationale (Informative) volume.

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

The Open Group
Apex Plaza, Forbury Road, Reading, Berkshire RG1 1AX, UK

Published 14 June 2024 by IEEE in the United States of America.
PDF: ISBN 979-8-8557-0793-9 STD26978
Print: ISBN 979-8-8557-0794-6 STDPD26978

Published 14 June 2024 by The Open Group in the United Kingdom
Doc. Number: C243
ISBN: 1-957866-40-6

This release of this standard is dedicated to the memory of Jörg Schilling and Donn Terry.

This standard has been prepared by the Austin Group. Feedback relating to the material contained within this standard may be submitted by using the Austin Group web site at www.opengroup.org/austin/defectform.html.

The following areas are outside the scope of POSIX.1-2024:

- Graphics interfaces
- Database management system interfaces
- Record I/O considerations
- Object or binary code portability
- System configuration and resource availability

POSIX.1-2024 describes the external characteristics and facilities that are of importance to application developers, rather than the internal construction techniques employed to achieve these capabilities. Special emphasis is placed on those functions and facilities that are needed in a wide variety of commercial applications.

Keywords: application program interface (API), argument, asynchronous, basic regular expression (BRE), built-in utility, byte, child, command language interpreter, CPU, extended regular expression (ERE), FIFO, file access control mechanism, IEEE 1003.1™, input/output (I/O), job control, network, parent, portable operating system interface (POSIX™), shell, stream, string, synchronous, system, thread, X/Open System Interface (XSI)

The Open Group

The Open Group is a global consortium that enables the achievement of business objectives through technology standards and open standards by fostering a culture of collaboration, inclusivity, and mutual respect among our diverse membership of more than 900 organizations. Our membership includes customers, systems and solutions suppliers, tools vendors, integrators, academics, and consultants across multiple industries.

The mission of The Open Group is to drive the creation of Boundaryless Information Flow™ achieved by:

- Working with customers to capture, understand, and address current and emerging requirements, establish policies, and share best practices
- Working with suppliers, consortia, and standards bodies to develop consensus and facilitate interoperability, to evolve and integrate specifications and open source technologies
- Offering a comprehensive set of services to enhance the operational efficiency of consortia
- Developing and operating the industry's premier certification service and encouraging procurement of certified products

Further information on The Open Group is available at https://www.opengroup.org.

The Open Group publishes a wide range of technical documentation, most of which is focused on development of Standards and Guides, but which also includes white papers, technical studies, certification and testing documentation, and business titles. Full details and a catalog are available at https://www.opengroup.org/library.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (https://standards.ieee.org/ipr/disclaimers.html), appear in all IEEE standards and may be found under the heading "Important Notices and Disclaimers Concerning IEEE Standards Documents."

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers involved in technical working groups are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning all standards, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement IEEE Standards documents do not guarantee safety, security, health, or environmental protection, or compliance with law, or guarantee against interference with or from other devices or networks. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE Standards documents are supplied "AS IS" and "WITH ALL FAULTS."

Use of an IEEE standard is wholly voluntary. The existence of an IEEE standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document should rely upon their own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus balloting process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English language version published by IEEE is the approved IEEE standard.

Use by artificial intelligence systems

In no event shall material in this document be used for the purpose of creating, training, enhancing, developing, maintaining, or contributing to any artificial intelligence systems without the express, written consent of IEEE SA and The Open Group in advance. “Artificial intelligence” refers to any software, application, or other system that uses artificial intelligence, machine learning, or similar technologies, to analyze, train, process, or generate content. Requests for consent can be submitted by email to austin-group-permissions@opengroup.org.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual is not, and shall not be considered or inferred to be, the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE or IEEE SA. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter’s views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s). News releases about IEEE standards issued by entities other than IEEE SA should be considered the view of the entity issuing the release rather than the formal position of IEEE or IEEE SA.

Comments on standards

Feedback relating to the material contained within this standard may be submitted by using the Austin Group web site at http://www.opengroup.org/austin/defectform.html.

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

Photocopies

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit IEEE Xplore or contact IEEE.[1] For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the IEEE SA Website.[2] Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in IEEE Xplore. Users are encouraged to periodically check for errata.

[1] Available at: https://ieeexplore.ieee.org/browse/standards/collection/ieee.
[2] Available at: https://standards.ieee.org/standard/index.html.

Patents

IEEE standards are developed in compliance with the IEEE SA Patent Policy.[3]

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at https://standards.ieee.org/about/sasb/patcom/patents.html. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

IMPORTANT NOTICE

Technologies, application of technologies, and recommended procedures in various industries evolve over time. The IEEE standards development process allows participants to review developments in industries, technologies, and practices, and to determine what, if any, updates should be made to the IEEE standard. During this evolution, the technologies and recommendations in IEEE standards may be implemented in ways not foreseen during the standard's development. IEEE standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, data privacy, and interference protection practices and all applicable laws and regulations.

[3] Available at: https://standards.ieee.org/about/sasb/patcom/materials.html.

Participants

IEEE Std 1003.1™-2024 was prepared by the Austin Group, sponsored by the Microprocessor Standards Committee of the IEEE Computer Society, The Open Group, and ISO/IEC JTC 1/SC22.

The Austin Group

At the time this IEEE standard was completed, the Austin Group had the following membership:

Andrew Josey, Chair
Donald W. Cragun, Organizational Representative, IEEE MSC
Nicholas M. Stoughton, Organizational Representative, ISO/IEC JTC 1/SC22
Eric Blake, Organizational Representative, The Open Group
Cathy Fox, Geoff Clare, Technical Editors

Austin Group Technical Reviewers

William Ahern
Mohamed Akram
Joe Auricchio
Ori Avtalion
Bogdan Barbu
Steve Bartolomei
Petr Baudis
Fabrice Bauzac
Eric Blake
Mark S. Brown
Erik Cederstrand
Stéphane Chazelas
Scott Cheloha
Alexander Cherepanov
Geoff Clare
Robert Clausecker
Daniel Colascione
Garrett Cooper
Alan Coopersmith
Ralph Corderoy
Ciprian Dorin Craciun
Donald W. Cragun
Mike Crowe
Martijn Dekker
Andrés Delfino
D.J. Delorie
Matthew Dempsky
Antonio Diaz
Ulrich Drepper
Paul Eggert
Robert Elz
Steve Emmerson
Laszlo Ersek
Andras Farkas
Richard Felker
Dirk Fieldhouse
Mike Frysinger
Mark Galeck
Enrique Garcia
Thorsten Glaser
Dmitry Goncharov
Christopher M. Graff
Quinn Grier
Philip Guenther
Bruno Haible
Richard Hansen
Guy Harris
Mark Harris
Gavin Howard
Elliott Hughes
Roland Illig
Jarmo Jaakkola
Andrew Josey
Nickolas Raymond Kaczynski
Nate Karstens
Michael Kerrisk
Alexey Khoroshilov
Elad Lahav
Jeff Layton
Vincent Lefèvre
Mark Lundblad
Roger Marquis
Nikos Mavrogiannopoulos
Davin McCall
Mihail Mihaylov
Todd C. Miller
Christoph Anton Mitterer
Mihai Moldovan
Ed Morton
Joseph S. Myers
Szabolcs Nagy
Jonathan Nieder
Danny Niu
Steffen Nurpmeso
Richard Palethorpe
Daniele Palumbo
Isabella Parakiss
Ben Pfaff
J. William Piggott
Wayne Pollock
Quentin Rameau
Martin Řehák
Torvald Riegel
G. Branden Robinson
Xavier Roche
Bastien Roucaries
Daniel Sabogal
Askar Safin
Jörg Schilling
Ed Schouten
Konrad Schwarz
Ingo Schwarze
Martin Sebor
Olaf 'Rhialto' Seibert
Joel Sherrill
Curtis Smith
Paul Smith
Job Snijders
Oliver Soong
Dimitri Staessens
Nicholas M. Stoughton
Sören Tempel
Jilles Tjoelker
William Toth
Fred J. Tydeman
Stijn van Dronrgelen
Lawrence Velázquez
Evgeny Vereshchagin
Rasmus Villemoes
Dennis Wölfing
Jonathan Wakely
Colin Watson
Nathan Weeks
Florian Weimer
Zack Weinberg
David A. Wheeler
Nicolas Williams
Yousong Zhou
Mark Ziegast
Roman Žilka

Austin Group Working Group Members

Hans Aberg
Eric Ackermann
Godmar Back
Eric Blake
Volodymyr Boyko
Andries E. Brouwer
Mark S. Brown
Jefferson Carpenter
Olivier Certner
Stéphane Chazelas
Tom Cherry
Earl Chew
Geoff Clare
Joshua M. Clulow
Alan Coopersmith
Donald W. Cragun
Mike Crowe
Martijn Dekker
Matthew Dempsky
Drew DeVault
Casper Dik
Deepa Dinamani
Dan Douglas
Niall Douglas
Ulrich Drepper
Lawrence D.K.B. Dwyer
Paul Eggert
Daniel Eischen
Julian Elischer
Robert Elz
Bruce Evans
Richard Felker
Jeffrey K. Fellin
Dirk Fieldhouse
Hal Finkel
Michael Forney
Mike Frysinger
Mark Galeck
Thorsten Glaser
Andreas Grapentin
Michael Greenberg
Philip Guenther
Joseph M. Gwinn
Jan Hafer
Bruno Haible
Richard Hansen
Mark Harris
David Holland
Gavin Howard
Elliott Hughes
Roland Illig
Lennart Jablonka
Chris F.A. Johns
Darrin Johnson
Andrew Josey
Nate Karstens
Dan Kegel
Michael Kerrisk
Anton Khikhlukha
Ukko Koknevics
Bruce Korb
David Korn
Rob Landley
Vincent Lefèvre
Wojtek Lerch
Charlie Lin
Scott Lurndal
Roger Marquis
Davin McCall
Stephen Michell
Per Mildner
Christoph Anton Mitterer
Thomas Mueller
Wilhelm Mueller
Koichi Murase
Joseph S. Myers
Danny Niu
Gian Ntzik
Steffen Nurpmeso
Carlos O'Donell
Andrew Pennebaker
Steven Penny
Colin Percival
J. William Piggott
Wayne Pollock
Quentin Rameau
Chet Ramey
Gabriel Ravier
G. Branden Robinson
Eric Sanchis
Daniel Santos
Jörg Schilling
Ed Schouten
Konrad Schwarz
Ingo Schwarze
John Scott
Simon Ser
Joel Sherrill
Thor Lancelot Simon
Keld Simonsen
Paul Smith
Job Snijders
Gabriel Soldani
Oliver Soong
Dimitri Staessens
Marc J. Stephenson
Nicholas M. Stoughton
Oskar Sveinsen
Alfred M. Szmidt
Tapani Tarvainen
Alexander Terekhov
Donn Terry
Jilles Tjoelker
Fred J. Tydeman
Oğuz Uysal
Harald van Dijk
Lawrence Velázquez
Oleksii Vilchansk
Corinna Vinschen
Jonathan Wakely
L.A. Walsh
David A. Wheeler
Jakub Wilk
Dennis Wölfing
Garrett Wollman
Jörg Wunsch
Ryan Zezeski
Mark Ziegast
Jason Zions

The Open Group

When The Open Group approved the Base Specifications, Issue 8, (technically identical to this standard) on 21 March 2024, the membership of The Open Group Base Working Group was as follows:

Andrew Josey, Chair
Eric Blake, Austin Group Liaison
Cathy Fox, Geoff Clare, Technical Editor

Base Working Group Members

Joe Auricchio
Eric Blake
Geoff Clare
Donald W. Cragun
Andrew Josey
Mark Ziegast

IEEE

At the time this standard was completed, the Microprocessor Committee had the following membership:

Ralph Baker Kearfott, *Chair*
Leonard Tsai, *Vice Chair and P754 Chair*
Andrew Josey, *P1003.1 Chair*
Donald W. Cragun, *Austin Group Liaison*
Joseph M. Gwinn, *Ex-officio Emeritus*
Richard Bugg, *P1722.1 Chair*
Kiran Gunnam, *P3109 Chair*
David Hough, *Outgoing P754 Chair*
Dave Olsen, *P1722 Chair*
Nathalie Revol, *P1788 Chair*
Blaise Vignon, *P3109 Chair*

The following members of the individual Standards Association balloting group voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Boon Chong Ang
Steven Bezner
Diego Chiozzi
Donald W. Cragun
Andrew Fieldsend
David Fuschi
Jie Guan
Joseph M. Gwinn
Werner Hoelzl
Andrew Josey
Piotr Karocki
Kenneth Lang
Rajesh Murthy
Venkatesha Prasad
Stephen Schwarm
Walter Struppler
Oren Yuen
Janusz Zalewski

When the IEEE SA Standards Board approved this standard on 20 May 2024, it had the following membership:

David J. Law, *Chair*
Jon Walter Rosdahl, *Vice Chair*
Gary Hoffman, *Past Chair*
Alpesh Shah, *Secretary*

Sara R. Biyabani
Ted Burse
Stephen Dukes
Doug Edwards
J. Travis Griffith
Guido R. Hiertz
Ronald W Hotchkiss
Hao Hu
Yousef Kimiagar
Joseph L. Koepfinger*
Howard Li
Xiaohui Liu
John Haiying Lu
Kevin W. Lu
Hiroshi Mano
Paul Nikolich
Robby Robson
Lei Wang
F. Keith Waters
Sha Wei
Philip B. Winston
Don Wright

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1003.1™-2024, IEEE Standard for Information Technology—Portable Operating System Interface (POSIX™)—Base Specifications, Issue 8.

This draft standard was developed, and is maintained, by a joint working group of members of the IEEE Microprocessor Standards Committee, members of The Open Group, and members of ISO/IEC Joint Technical Committee 1. This joint working group is known as the Austin Group.[4]

The Austin Group arose out of discussions amongst the parties which started in early 1998, leading to an initial meeting and formation of the group in September 1998. The purpose of the Austin Group is to develop and maintain the core open systems interfaces that are the POSIX 1003.1 (and former 1003.2) standards, ISO/IEC 9945, and the core of the Single UNIX® Specification.

The approach to specification development has been one of "write once, adopt everywhere", with the deliverables being a set of specifications that carry the IEEE POSIX designation, The Open Group Standard designation, and an ISO/IEC designation.

This unique development has combined both the industry-led efforts and the formal standardization activities into a single initiative, and included a wide spectrum of participants. The Austin Group continues as the maintenance body for this document.

Anyone wishing to participate in the Austin Group should contact the chair with their request. There are no fees for participation or membership. You may participate as an observer or as a contributor. You do not have to attend face-to-face meetings to participate; electronic participation is most welcome. For more information on the Austin Group and how to participate, see www.opengroup.org/austin.

Background

The developers of POSIX.1-2024 represent a cross-section of hardware manufacturers, vendors of operating systems and other software development tools, software designers, consultants, academics, authors, applications programmers, and others.

Conceptually, POSIX.1-2024 describes a set of fundamental services needed for the efficient construction of application programs. Access to these services has been provided by defining an interface, using the C programming language, a command interpreter, and common utility programs that establish standard semantics and syntax. Since this interface enables application developers to write portable applications – it was developed with that goal in mind – it has been designated POSIX,[5] an acronym for Portable Operating System Interface.

Although originated to refer to the original IEEE Std 1003.1-1988, the name POSIX more correctly refers to a *family* of related standards: IEEE Std 1003.*n* and the parts of ISO/IEC 9945. In earlier editions of the IEEE Standard, the term POSIX was used as a synonym for IEEE Std 1003.1-1988. A preferred term, POSIX.1, emerged. This maintained the advantages of readability of the symbol "POSIX" without being ambiguous with the POSIX family of standards.

[4] The Austin Group is named after the location of the inaugural meeting held at the IBM facility in Austin, Texas in September 1998.

[5] The name POSIX was suggested by Richard Stallman. It is expected to be pronounced with the first two syllables as in positive, not poh-six, or other variations. The pronunciation has been published in an attempt to promulgate a standardized way of referring to a standard operating system interface.

Audience

The intended audience for POSIX.1-2024 is all persons concerned with an industry-wide standard operating system based on the UNIX system. This includes at least four groups of people:

- Persons buying hardware and software systems
- Persons managing companies that are deciding on future corporate computing directions
- Persons implementing operating systems, and especially
- Persons developing applications where portability is an objective

Purpose

Several principles guided the development of POSIX.1-2024:

- Application-Oriented – The basic goal was to promote portability of application programs across UNIX system environments by developing a clear, consistent, and unambiguous standard for the interface specification of a portable operating system based on the UNIX system documentation. POSIX.1-2024 codifies the common, existing definition of the UNIX system.
- Interface, Not Implementation – POSIX.1-2024 defines an interface, not an implementation. No distinction is made between library functions and system calls; both are referred to as functions. No details of the implementation of any function are given (although historical practice is sometimes indicated in the RATIONALE section). Symbolic names are given for constants (such as signals and error numbers) rather than numbers.
- Source, Not Object, Portability – POSIX.1-2024 has been written so that a program written and translated for execution on one conforming implementation may also be translated for execution on another conforming implementation. POSIX.1-2024 does not guarantee that executable (object or binary) code will execute under a different conforming implementation than that for which it was translated, even if the underlying hardware is identical.
- The C Language – The system interfaces and header definitions are written in terms of the standard C language as specified in the ISO C standard.
- No Superuser, No System Administration – There was no intention to specify all aspects of an operating system. System administration facilities and functions are excluded from this standard, and functions usable only by the superuser have not been included. Still, an implementation of the standard interface may also implement features not in POSIX.1-2024. POSIX.1-2024 is also not concerned with hardware constraints or system maintenance.
- Minimal Interface, Minimally Defined – In keeping with the historical design principles of the UNIX system, the mandatory core facilities of POSIX.1-2024 have been kept as minimal as possible. Additional capabilities have been added as optional extensions.

- Broadly Implementable – The developers of POSIX.1-2024 endeavored to make all specified functions implementable across a wide range of existing and potential systems, including:
    - All of the current major systems that are ultimately derived from the original UNIX system code (Version 7 or later)
    - Compatible systems that are not derived from the original UNIX system code
    - Emulations hosted on entirely different operating systems
    - Networked systems
    - Distributed systems

— Systems running on a broad range of hardware

No direct references to this goal appear in POSIX.1-2024, but some results of it are mentioned in the Rationale (Informative) volume.

- Minimal Changes to Historical Implementations – When the original version – IEEE Std 1003.1-1988 – was published, there were no known historical implementations that did not have to change. However, there was a broad consensus on a set of functions, types, definitions, and concepts that formed an interface that was common to most historical implementations.

  The adoption of the 1988 and 1990 IEEE system interface standards, the 1992 IEEE shell and utilities standard, the various The Open Group (formerly X/Open) specifications, and IEEE Std 1003.1-2001 and its technical corrigenda have consolidated this consensus, and this version reflects the significantly increased level of consensus arrived at since the original versions. The authors of the original versions tried, as much as possible, to follow the principles below when creating new specifications:

  — By standardizing an interface like one in an historical implementation; for example, directories

  — By specifying an interface that is readily implementable in terms of, and backwards-compatible with, historical implementations, such as the extended tar format defined in the pax utility

  — By specifying an interface that, when added to an historical implementation, will not conflict with it; for example, the sigaction() function

  POSIX.1-2024 is specifically not a codification of a particular vendor's product.

  It should be noted that implementations will have different kinds of extensions. Some will reflect "historical usage" and will be preserved for execution of pre-existing applications. These functions should be considered "obsolescent" and the standard functions used for new applications. Some extensions will represent functions beyond the scope of POSIX.1-2024. These need to be used with careful management to be able to adapt to future extensions of POSIX.1-2024 and/or port to implementations that provide these services in a different manner.

- Minimal Changes to Existing Application Code—A goal of POSIX.1-2024 was to minimize additional work for application developers. However, because every known historical implementation will have to change at least slightly to conform, some applications will have to change.

POSIX.1-2024

POSIX.1-2024 defines the Portable Operating System Interface (POSIX) requirements and consists of the following topics arranged as a series of volumes within the standard:

- Base Definitions
- System Interfaces
- Shell and Utilities
- Rationale (Informative)

Base Definitions

The Base Definitions volume provides common definitions for this standard, therefore readers should be familiar with it before using the other volumes.

This volume is structured as follows:

- Chapter 1 is an introduction.
- Chapter 2 defines the conformance requirements.
- Chapter 3 defines general terms used.
- Chapter 4 describes general concepts used.
- Chapter 5 describes the notation used to specify file input and output formats in this volume and the Shell and Utilities volume.
- Chapter 6 describes the portable character set and the process of character set definition.
- Chapter 7 describes the syntax for defining internationalization locales as well as the POSIX locale provided on all systems.
- Chapter 8 describes the use of environment variables for internationalization and other purposes.
- Chapter 9 describes the syntax of pattern matching using regular expressions employed by many utilities and matched by the *regcomp*() and *regexec*() functions.
- Chapter 10 describes files and devices found on all systems.
- Chapter 11 describes the asynchronous terminal interface for many of the functions in the System Interfaces volume and the *stty* utility in the Shell and Utilities volume.
- Chapter 12 describes the policies for command line argument construction and parsing.
- Chapter 13 describes namespace reservation.
- Chapter 14 defines the contents of headers which declare the functions and global variables, and define types, constants, macros, and data structures that are needed by programs using the services provided by the System Interfaces volume.

Comprehensive references are available in the Index.

System Interfaces

The System Interfaces volume describes the interfaces offered to application programs by POSIX-conformant systems. Readers are expected to be experienced C language programmers, and to be familiar with the Base Definitions volume.

This volume is structured as follows:

- Chapter 1 explains the status of this volume and its relationship to other formal standards.
- Chapter 2 contains important concepts, terms, and caveats relating to the rest of this volume.
- Chapter 3 defines the functional interfaces to the POSIX-conformant system.

Comprehensive references are available in the Index.

Shell and Utilities

The Shell and Utilities volume describes the commands and utilities offered to application programs on POSIX-conformant systems. Readers are expected to be familiar with the Base Definitions volume.

This volume is structured as follows:

- Chapter 1 explains the status of this volume and its relationship to other formal standards. It also describes the defaults used by the utility descriptions.
- Chapter 2 describes the command language used in POSIX-conformant systems, and special built-in utilities.
- Chapter 3 consists of reference pages for all utilities, other than the special built-in utilities described in Chapter 2, available on POSIX-conformant systems.

Comprehensive references are available in the Index.

Rationale (Informative)

The Rationale volume is published to assist in the process of review. It contains historical information concerning the contents of this standard and why features were included or discarded by the standard developers. It also contains notes of interest to application programmers on recommended programming practices, emphasizing the consequences of some aspects of POSIX.1-2024 that may not be immediately apparent.

This volume is organized in parallel to the normative volumes of this standard, with a separate part for each of the three normative volumes.

Within this volume, the following terms are used:

- Base standard—The portions of POSIX.1-2024 that are not optional, equivalent to the definitions of classic POSIX.1 and POSIX.2.
- POSIX.0—Although this term is not used in the normative text of POSIX.1-2024, it is used in this volume to refer to IEEE Std 1003.0-1995.
- POSIX.1b—Although this term is not used in the normative text of POSIX.1-2024, it is used in this volume to refer to the elements of the POSIX Realtime Extension amendment. (This was earlier referred to as POSIX.4 during the standard development process.)
- POSIX.1c—Although this term is not used in the normative text of POSIX.1-2024, it is used in this volume to refer to the POSIX Threads Extension amendment. (This was earlier referred to as POSIX.4a during the standard development process.)
- Standard developers—The individuals and companies in the development organizations responsible for POSIX.1-2024: the IEEE P1003.1 working groups, The Open Group Base working group, advised by the hundreds of individual technical experts who balloted the draft standards within the Austin Group, and the member bodies and technical experts of ISO/IEC JTC 1/SC 22.
- XSI option—The portions of POSIX.1-2024 addressing the extension added for support of the Single UNIX Specification.

Typographical Conventions

The following typographical conventions are used throughout this standard. In the text, this standard is referred to as POSIX.1-2024, which is technically identical to The Open Group Base Specifications, Issue 8.

The typographical conventions listed here are for ease of reading only. Editorial inconsistencies in the use of typography are unintentional and have no normative meaning in POSIX.1-2024.

Reference	Example	Notes
C-Language Data Structure	**aiocb**	

Reference	Example	Notes
C-Language Data Structure Member	*aio_lio_opcode*	
C-Language Data Type	**long**	
C-Language External Variable	*errno*	
C-Language Function	*system*()	
C-Language Function Argument	*arg*	
C-Language Function Family	*exec*	
C-Language Header	**<sys/stat.h>**	
C-Language Keyword	**return**	
C-Language Macro with Argument	*assert*()	
C-Language Macro with No Argument	NET_ADDRSTRLEN	
C-Language Preprocessing Directive	**#define**	
Commands within a Utility	**a**, **c**	
Conversion Specifier, Specifier/Modifier Character	`%A`, `g`, `E`	1
Environment Variable	*PATH*	
Error Number	[EINTR]	
Example Output	**`Hello, World`**	
Filename	**/tmp**	
Literal Character	`'c'`, `'\r'`	2
Literal String	`"abcde"`	2
Optional Items in Utility Syntax	**[ ]**	
Parameter	*<directory pathname>*	
Special Character	<newline>	3
Symbolic Constant	_POSIX_VDISABLE	
Symbolic Limit, Configuration Value	{LINE_MAX}	4
Syntax	`#include <sys/stat.h>`	
User Input and Example Code	`echo Hello, World`	5
Utility Name	*awk*	
Utility Operand	*file_name*	
Utility Option	**-c**	
Utility Option with Option-Argument	**-w** *width*	

Note that:

1. Conversion specifications, specifier characters, and modifier characters are used primarily in date-related functions and utilities and the *fprintf*() and *fscanf*() formatting functions.

2. Unless otherwise noted, the quotes shall not be used as input or output. When used in a list item, the quotes are omitted. The literal characters <apostrophe> (also known as single-quote) and <backslash> are either shown as the C constants `'\''` and `'\\'`, respectively, or as the special characters <apostrophe>, single-quote, and <backslash> depending on context.

3. The style selected for some of the special characters, such as <newline>, matches the form of the input given to the *localedef* utility. Generally, the characters selected for this special treatment are those that are not visually distinct, such as the control characters <tab> or <newline>.

4. Names surrounded by braces represent symbolic limits or configuration values which may be declared in appropriate headers by means of the C **#define** construct, or obtained at runtime from functions or utilities that return limit or configuration values.

5. Brackets shown in this font, `"[ ]"`, are part of the syntax and do not indicate optional items. In syntax the `'|'` symbol is used to separate alternatives, and ellipses (`"..."`) are used to show that additional arguments are optional.

Shading is used to identify extensions and options.

Footnotes and notes within the body of the normative text are for information only (informative).

Informative sections (such as Rationale, Change History, Application Usage, and so on) are denoted by continuous shading bars in the margins.

Ranges of values are indicated with parentheses or brackets as follows:

1. (*a*,*b*) means the range of all values from *a* to *b*, including neither *a* nor *b*.

2. [*a*,*b*] means the range of all values from *a* to *b*, including *a* and *b*.

3. [*a*,*b*) means the range of all values from *a* to *b*, including *a*, but not *b*.

4. (*a*,*b*] means the range of all values from *a* to *b*, including *b*, but not *a*.

NOTE— A symbolic limit beginning with POSIX is treated differently, depending on context. In a C-language header, the symbol POSIXstring (where string may contain underscores) is represented by the C identifier _POSIXstring, with a leading underscore required to prevent ISO C standard name space pollution. However, in other contexts, such as languages other than C, the leading underscore is not used because this requirement does not exist.

Contents

List of Figures

List of Tables

Contents

Trademarks

The following information is given for the convenience of users of POSIX.1-2024 and does not constitute an endorsement by the IEEE or The Open Group of these products. Equivalent products may be used if they can be shown to lead to the same results.

There may be other products mentioned in the text that might be covered by trademark protection and readers are advised to verify them independently.

AIX® and IBM® are registered trademarks of International Business Machines Corporation.

ArchiMate®, FACE®, FACE® logo, Future Airborne Capability Environment®, Making Standards Work®, Open Footprint®, Open O® logo, Open O and Check® certification logo, OSDU®, Platform 3.0®, The Open Group®, TOGAF®, UNIX®, UNIXWARE®, and X® logo are registered trademarks and Boundaryless Information Flow™, Build with Integrity Buy with Confidence™, Commercial Aviation Reference Architecture™, Dependability Through Assuredness™, Digital Practitioner Body of Knowledge™, DPBoK™, EMMM™, FHIM Profile Builder™, FHIM logo, FPB™, IT4IT™, IT4IT™ logo, O-AA™, O-DA™, O-DEF™, O-HERA™, O-PAS™, O-TTPS™, Open Agile Architecture™, Open FAIR™, Open Process Automation™, Open Subsurface Data Universe™, Open Trusted Technology Provider™, Sensor Integration Simplified™, Sensor Open Systems Architecture™, SOSA™, and SOSA™ logo are trademarks of The Open Group.

AT&T® is a registered trademark of AT&T in the USA and other countries.

BSD™ is a trademark of the University of California, Berkeley, USA.

Hewlett Packard®, HP®, and HP-UX® are registered trademarks of HP Hewlett Packard Group LLC.

IEEE® is a registered trademark, and POSIX™, 754™, 854™, 1003.0™, 1003.1™, 1003.1d™, 1003.1g™, 1003.1j™, 1003.1q™, 1003.2™, 1003.2a™, 1003.2d™, 1003.9™, and 1003.13™ are trademarks of The Institute of Electrical and Electronic Engineers, Inc.

Linux® is a registered trademark of Linus Torvalds.

Sun® and Sun Microsystems® are registered trademarks of Oracle America, Inc.

/usr/group® is a registered trademark of UniForum, the International Network of UNIX System Users.

Acknowledgements

The contributions of the following organizations to the development of POSIX.1-2024 are gratefully acknowledged:

- AT&T for permission to reproduce portions of its copyrighted System V Interface Definition (SVID) and material from the UNIX System V Release 2.0 documentation
- Hewlett-Packard Company, International Business Machines Corporation, Novell Inc., The Open Software Foundation, and Sun Microsystems Inc. for permission to reproduce portions of their copyrighted documentation
- ISO/IEC JTC 1/SC 22/WG 14 C Language Committee
- Red Hat Inc. for permission to reproduce portions of its copyrighted documentation

POSIX.1-2024 was prepared by the Austin Group, a joint working group of the IEEE, The Open Group, and ISO/IEC JTC 1/SC 22.

Referenced Documents

Normative References

Normative references for POSIX.1-2024 are defined in Section 1.4 (on page 5).

Informative References

The following documents are referenced in POSIX.1-2024:

1984 /usr/group Standard
/usr/group Standards Committee, Santa Clara, CA, UniForum 1984.

Almasi and Gottlieb
George S. Almasi and Allan Gottlieb, *Highly Parallel Computing*, The Benjamin/Cummings Publishing Company, Inc., 1989, ISBN: 0-8053-0177-1.

ANSI C
American National Standard for Information Systems: Standard X3.159-1989, Programming Language C.

ANSI X3.226-1994
American National Standard for Information Systems: Standard X3.226-1994, Programming Language Common LISP.

Brawer
Steven Brawer, *Introduction to Parallel Programming*, Academic Press, 1989, ISBN: 0-12-128470-0.

DeRemer and Pennello Article
DeRemer, Frank and Pennello, Thomas J., *Efficient Computation of LALR(1) Look-Ahead Sets*, SigPlan Notices, Volume 15, No. 8, August 1979.

Draft ANSI X3J11.1
IEEE Floating Point draft report of ANSI X3J11.1 (NCEG).

FIPS 151-1
Federal Information Procurement Standard (FIPS) 151-1. Portable Operating System Interface (POSIX)—Part 1: System Application Program Interface (API) [C Language].

FIPS 151-2
Federal Information Procurement Standards (FIPS) 151-2, Portable Operating System Interface (POSIX)— Part 1: System Application Program Interface (API) [C Language].

HP-UX Manual
Hewlett-Packard HP-UX Release 9.0 Reference Manual, Third Edition, August 1992.

IEC 60559: 1989
IEC 60559: 1989, Binary Floating-Point Arithmetic for Microprocessor Systems (previously designated IEC 559: 1989).

IEEE Standards Terms
IEEE 100, The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition.

IEEE Std 754™-1985
IEEE Std 754-1985 (Reaff 1990), IEEE Standard for Binary Floating-Point Arithmetic.

IEEE Std 854™-1987
IEEE Std 854-1987, IEEE Standard for Radix-Independent Floating-Point Arithmetic.

IEEE Std 1003.9™-1992
IEEE Std 1003.9-1992, IEEE Standard for Information Technology — POSIX FORTRAN 77 Language Interfaces — Part 1: Binding for System Application Program Interface API.

IETF RFC 791
Internet Protocol, Version 4 (IPv4), September 1981 (available at: www.ietf.org/rfc/rfc0791.txt).

IETF RFC 819
The Domain Naming Convention for Internet User Applications, Z. Su, J. Postel, August 1982 (available at: www.ietf.org/rfc/rfc0819.txt).

IETF RFC 919
Broadcasting Internet Datagrams, J. Mogul, October 1984 (available at: www.ietf.org/rfc/rfc0919.txt).

IETF RFC 920
Domain Requirements, J. Postel, J. Reynolds, October 1984 (available at: www.ietf.org/rfc/rfc0920.txt).

IETF RFC 921
Domain Name System Implementation Schedule, J. Postel, October 1984 (available at: www.ietf.org/rfc/rfc0921.txt).

IETF RFC 922
Broadcasting Internet Datagrams in the Presence of Subnets, J. Mogul, October 1984 (available at: www.ietf.org/rfc/rfc0922.txt).

IETF RFC 1034
Domain Names — Concepts and Facilities, P. Mockapetris, November 1987 (available at: www.ietf.org/rfc/rfc1034.txt).

IETF RFC 1035
Domain Names — Implementation and Specification, P. Mockapetris, November 1987 (available at: www.ietf.org/rfc/rfc1035.txt).

IETF RFC 1123
Requirements for Internet Hosts — Application and Support, R. Braden, October 1989 (available at: www.ietf.org/rfc/rfc1123.txt).

IETF RFC 1951
DEFLATE Compressed Data Format Specification version 1.3, P. Deutsch, May 1996 (available at: www.ietf.org/rfc/rfc1951.txt).

IETF RFC 1952
GZIP file format specification version 4.3, P. Deutsch, May 1996 (available at: www.ietf.org/rfc/rfc1952.txt).

IETF RFC 2045
Multipurpose Internet Mail Extensions (MIME), Part 1: Format of Internet Message Bodies, N. Freed, N. Borenstein, November 1996 (available at: www.ietf.org/rfc/rfc2045.txt).

IETF RFC 2181
: Clarifications to the DNS Specification, R. Elz, R. Bush, July 1997 (available at: www.ietf.org/rfc/rfc2181.txt).

IETF RFC 3596
: DNS Extensions to Support IP Version 6, S. Thomson, C. Huitema, V. Ksinant, M. Souissi, October 2003 (available at: www.ietf.org/rfc/rfc3596.txt).

IETF RFC 4291
: IP Version 6 Addressing Architecture, R. Hinden, S. Deering, February 2006 (available at: www.ietf.org/rfc/rfc4291.txt).

IETF RFC 5322
: Internet Message Format, P. Resnick, October 2008 (available at: www.ietf.org/rfc/rfc5322.txt).

IETF RFC 6557
: Procedures for Maintaining the Time Zone Database, E. Lear, P. Eggert, February 2012 (available at: www.ietf.org/rfc/rfc6557.txt).

IETF RFC 8200
: Internet Protocol, Version 6 (IPv6) Specification, S. Deering, R. Hinden, July 2017 (available at: www.ietf.org/rfc/rfc8200.txt).

Internationalisation Guide
: Guide, July 1993, Internationalisation Guide, Version 2 (ISBN: 1-859120-02-4, G304), published by The Open Group.

ISO 2375: 1985
: ISO 2375: 1985, Data Processing — Procedure for Registration of Escape Sequences.

ISO 8652: 1987
: ISO 8652: 1987, Programming Languages — Ada (technically identical to ANSI standard 1815A-1983).

ISO/IEC 1539: 1991
: ISO/IEC 1539: 1991, Information Technology — Programming Languages — Fortran (technically identical to the ANSI X3.9-1978 standard [FORTRAN 77]).

ISO/IEC 4873: 1991
: ISO/IEC 4873: 1991, Information Technology — ISO 8-bit Code for Information Interchange — Structure and Rules for Implementation.

ISO/IEC 6429: 1992
: ISO/IEC 6429: 1992, Information Technology — Control Functions for Coded Character Sets.

ISO/IEC 6937: 1994
: ISO/IEC 6937: 1994, Information Technology — Coded Graphic Character Set for Text Communication — Latin Alphabet.

ISO/IEC 8802-3: 1996
: ISO/IEC 8802-3: 1996, Information Technology — Telecommunications and Information Exchange Between Systems — Local and Metropolitan Area Networks — Specific Requirements — Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications.

ISO/IEC 8859
: ISO/IEC 8859, Information Technology — 8-Bit Single-Byte Coded Graphic Character Sets:

Part 1: Latin Alphabet No. 1
Part 2: Latin Alphabet No. 2
Part 3: Latin Alphabet No. 3
Part 4: Latin Alphabet No. 4
Part 5: Latin/Cyrillic Alphabet
Part 6: Latin/Arabic Alphabet
Part 7: Latin/Greek Alphabet
Part 8: Latin/Hebrew Alphabet
Part 9: Latin Alphabet No. 5
Part 10: Latin Alphabet No. 6
Part 11: Latin/Thai Alphabet
Part 13: Latin Alphabet No. 7
Part 14: Latin Alphabet No. 8 (Celtic)
Part 15: Latin Alphabet No. 9
Part 16: Latin Alphabet No. 10

ISO/IEC 9899: 1990
ISO/IEC 9899: 1990, Programming Languages — C, including Amendment 1: 1995 (E), C Integrity (Multibyte Support Extensions (MSE) for ISO C).

ISO POSIX-1: 1996
ISO/IEC 9945-1: 1996, Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language] (identical to ANSI/IEEE Std 1003.1-1996). Incorporating ANSI/IEEE Stds 1003.1-1990, 1003.1b-1993, 1003.1c-1995, and 1003.1i-1995.

ISO POSIX-2: 1993
ISO/IEC 9945-2: 1993, Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities (identical to ANSI/IEEE Std 1003.2™-1992, as amended by ANSI/IEEE Std 1003.2a™-1992).

Issue 1
X/Open Portability Guide, July 1985 (ISBN: 0-444-87839-4).

Issue 2
X/Open Portability Guide, January 1987:

- Volume 1: XVS Commands and Utilities (ISBN: 0-444-70174-5)
- Volume 2: XVS System Calls and Libraries (ISBN: 0-444-70175-3)

Issue 3
X/Open Specification, 1988, 1989, February 1992:

- Commands and Utilities, Issue 3 (ISBN: 1-872630-36-7, C211); this specification was formerly X/Open Portability Guide, Issue 3, Volume 1, January 1989, XSI Commands and Utilities (ISBN: 0-13-685835-X, XO/XPG/89/002)
- System Interfaces and Headers, Issue 3 (ISBN: 1-872630-37-5, C212); this specification was formerly X/Open Portability Guide, Issue 3, Volume 2, January 1989, XSI System Interface and Headers (ISBN: 0-13-685843-0, XO/XPG/89/003)
- Curses Interface, Issue 3, contained in Supplementary Definitions, Issue 3 (ISBN: 1-872630-38-3, C213), Chapters 9 to 14 inclusive; this specification was formerly X/Open Portability Guide, Issue 3, Volume 3, January 1989, XSI Supplementary Definitions (ISBN: 0-13-685850-3, XO/XPG/89/004)

- Headers Interface, Issue 3, contained in Supplementary Definitions, Issue 3 (ISBN: 1-872630-38-3, C213), Chapter 19, Cpio and Tar Headers; this specification was formerly X/Open Portability Guide Issue 3, Volume 3, January 1989, XSI Supplementary Definitions (ISBN: 0-13-685850-3, XO/XPG/89/004)

Issue 4

CAE Specification, July 1992, published by The Open Group:

- System Interface Definitions (XBD), Issue 4 (ISBN: 1-872630-46-4, C204)
- Commands and Utilities (XCU), Issue 4 (ISBN: 1-872630-48-0, C203)
- System Interfaces and Headers (XSH), Issue 4 (ISBN: 1-872630-47-2, C202)

Issue 4, Version 2

CAE Specification, August 1994, published by The Open Group:

- System Interface Definitions (XBD), Issue 4, Version 2 (ISBN: 1-85912-036-9, C434)
- Commands and Utilities (XCU), Issue 4, Version 2 (ISBN: 1-85912-034-2, C436)
- System Interfaces and Headers (XSH), Issue 4, Version 2 (ISBN: 1-85912-037-7, C435)

Issue 5

Technical Standard, February 1997, published by The Open Group:

- System Interface Definitions (XBD), Issue 5 (ISBN: 1-85912-186-1, C605)
- Commands and Utilities (XCU), Issue 5 (ISBN: 1-85912-191-8, C604)
- System Interfaces and Headers (XSH), Issue 5 (ISBN: 1-85912-181-0, C606)

Issue 6

Technical Standard, April 2004, published by The Open Group:

- Base Definitions (XBD), Issue 6 (ISBN: 1-931624-43-7, C046)
- System Interfaces (XSH), Issue 6 (ISBN: 1-931624-44-5, C047)
- Shell and Utilities (XCU), Issue 6 (ISBN: 1-931624-45-3, C048)

Knuth Article

Knuth, Donald E., *On the Translation of Languages from Left to Right*, Information and Control, Volume 8, No. 6, October 1965.

KornShell

Bolsky, Morris I. and Korn, David G., *The New KornShell Command and Programming Language*, March 1995, Prentice Hall.

MSE Working Draft

Working draft of ISO/IEC 9899:1990/Add3:Draft, Addendum 3 — Multibyte Support Extensions (MSE) as documented in the ISO Working Paper SC22/WG14/N205 dated 31 March 1992.

POSIX.0:1995

IEEE Std 1003.0™-1995, IEEE Guide to the POSIX Open System Environment (OSE) (identical to ISO/IEC TR 14252).

POSIX.1:1988

IEEE Std 1003.1™-1988, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language].

POSIX.1: 1990
IEEE Std 1003.1™-1990, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) [C Language].

POSIX.1a
P1003.1a, Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — (C Language) Amendment.

POSIX.1d: 1999
IEEE Std 1003.1d™-1999, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 4: Additional Realtime Extensions [C Language].

POSIX.1g: 2000
IEEE Std 1003.1g™-2000, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 6: Protocol-Independent Interfaces (PII).

POSIX.1j: 2000
IEEE Std 1003.1j™-2000, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 5: Advanced Realtime Extensions [C Language].

POSIX.1q: 2000
IEEE Std 1003.1q™-2000, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 1: System Application Program Interface (API) — Amendment 7: Tracing [C Language].

POSIX.2: 1992
IEEE Std 1003.2™-1992, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities.

POSIX.2b
P1003.2b, Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities — Amendment.

POSIX.2d: 1994
IEEE Std 1003.2d™-1994, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 2: Shell and Utilities — Amendment 1: Batch Environment.

POSIX.13: 1998
IEEE Std 1003.13™-1998, IEEE Standard for Information Technology — Standardized Application Environment Profile (AEP) — POSIX Realtime Application Support.

POSIX.26: 2003
IEEE Std 1003.26™-2003, IEEE Standard for Information Technology — Portable Operating System Interface (POSIX) — Part 26: Device Control Application Program Interface (API) [C Language].

Sarwate Article
Sarwate, Dilip V., *Computation of Cyclic Redundancy Checks via Table Lookup*, Communications of the ACM, Volume 30, No. 8, August 1988.

Sprunt, Sha, and Lehoczky
Sprunt, B., Sha, L., and Lehoczky, J.P., *Aperiodic Task Scheduling for Hard Real-Time Systems*, The Journal of Real-Time Systems, Volume 1, 1989, Pages 27-60.

SVID, Issue 1
: American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue 1; Morristown, NJ, UNIX Press, 1985.

SVID, Issue 2
: American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue 2; Morristown, NJ, UNIX Press, 1986.

SVID, Issue 3
: American Telephone and Telegraph Company, System V Interface Definition (SVID), Issue 3; Morristown, NJ, UNIX Press, 1989.

The AWK Programming Language
: Aho, Alfred V., Kernighan, Brian W., and Weinberger, Peter J., *The AWK Programming Language*, Reading, MA, Addison-Wesley 1988.

The C Programming Language
: Kernighan, Brian W. and Ritchie, Dennis M., *The C Programming Language*, Englewood Cliffs, NJ, Prentice Hall, 1st Edition (February 1978) ISBN 0-13-110163-3; 2nd Edition (March 1988) ISBN 0-13-110362-8.

UNIX Programmer's Manual
: American Telephone and Telegraph Company, *UNIX Time-Sharing System: UNIX Programmer's Manual*, 7th Edition, Murray Hill, NJ, Bell Telephone Laboratories, January 1979.

XNS, Issue 4
: CAE Specification, August 1994, Networking Services, Issue 4 (ISBN: 1-85912-049-0, C438), published by The Open Group.

XNS, Issue 5
: CAE Specification, February 1997, Networking Services, Issue 5 (ISBN: 1-85912-165-9, C523), published by The Open Group.

XNS, Issue 5.2
: Technical Standard, January 2000, Networking Services (XNS), Issue 5.2 (ISBN: 1-85912-241-8, C808), published by The Open Group.

X/Open Curses, Issue 4, Version 2
: CAE Specification, May 1996, X/Open Curses, Issue 4, Version 2 (ISBN: 1-85912-171-3, C610), published by The Open Group.

Yacc
: *Yacc: Yet Another Compiler Compiler*, Stephen C. Johnson, 1978.

Source Documents

Parts of the following documents were used to create the base documents for POSIX.1-2001:

AIX 3.2 Manual
: AIX Version 3.2 For RISC System/6000, Technical Reference: Base Operating System and Extensions, 1990, 1992 (Part No. SC23-2382-00).

OSF/1
: OSF/1 Programmer's Reference, Release 1.2 (ISBN: 0-13-020579-6).

OSF AES
: Application Environment Specification (AES) Operating System Programming Interfaces Volume, Revision A (ISBN: 0-13-043522-8).

Referenced Documents

System V Release 2.0

— UNIX System V Release 2.0 Programmer's Reference Manual (April 1984 - Issue 2).

— UNIX System V Release 2.0 Programming Guide (April 1984 - Issue 2).

System V Release 4.2

Operating System API Reference, UNIX® SVR4.2 (1992) (ISBN: 0-13-017658-3).

The Open Group Standard

Vol. 1:

Base Definitions, Issue 8

The Open Group
The Institute of Electrical and Electronics Engineers, Inc.

Chapter 1

Introduction

1.1 Scope

POSIX.1-2024 defines a standard operating system interface and environment, including a command interpreter (or ``shell''), and common utility programs to support applications portability at the source code level. It is intended to be used by both application developers and system implementors.

POSIX.1-2024 comprises four major components (each in an associated volume):

1. General terms, concepts, and interfaces common to all volumes of POSIX.1-2024, including utility conventions and C-language header definitions, are included in the Base Definitions volume of POSIX.1-2024.
2. Definitions for system service functions and subroutines, language-specific system services for the C programming language, function issues, including portability, error handling, and error recovery, are included in the System Interfaces volume of POSIX.1-2024.
3. Definitions for a standard source code-level interface to command interpretation services (a ``shell'') and common utility programs for application programs are included in the Shell and Utilities volume of POSIX.1-2024.
4. Extended rationale that did not fit well into the rest of the document structure, containing historical information concerning the contents of POSIX.1-2024 and why features were included or discarded by the standard developers, is included in the Rationale (Informative) volume of POSIX.1-2024.

The following areas are outside of the scope of POSIX.1-2024:

- Graphics interfaces
- Database management system interfaces
- Record I/O considerations
- Object or binary code portability
- System configuration and resource availability

POSIX.1-2024 describes the external characteristics and facilities that are of importance to application developers, rather than the internal construction techniques employed to achieve these capabilities. Special emphasis is placed on those functions and facilities that are needed in a wide variety of commercial applications.

The facilities provided in POSIX.1-2024 are drawn from the following base documents:

- IEEE Std 1003.1-2017 (POSIX.1-2017)
- IEEE Std 1003.26-2003 (POSIX.26-2003)

- ISO/IEC 9899: 2018, Programming Languages — C (C17)
- ISO/IEC TR 24731-2: 2010, Programming languages, their environments and system software interfaces — Extensions to the C library — Part 2: Dynamic Allocation Functions
- The Open Group Standard, 2021, Additional APIs for the Base Specifications Issue 8, Part 1
- The Open Group Standard, 2022, Additional APIs for the Base Specifications Issue 8, Part 2

Emphasis has been placed on standardizing existing practice for existing users, with changes and additions limited to correcting deficiencies in the following areas:

- Issues raised by Austin Group defect reports and IEEE Interpretations against IEEE Std 1003.1.
- Issues raised in corrigenda for The Open Group Standards and working group resolutions from The Open Group
- Changes to make the text self-consistent with the additional material merged
- Features, marked obsolescent in the base documents, have been considered for removal in this version
- Alignment with the ISO/IEC 9899: 2018 standard

1.2 Word Usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).[1, 2]

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may* equals *is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can* equals *is able to*).

1.3 Conformance

Conformance requirements for POSIX.1-2024 are defined in Chapter 2 (on page 15).

1. The use of the word *must* is deprecated and cannot be used when stating mandatory requirements; *must* is used only to describe unavoidable situations.
2. The use of *will* is deprecated and cannot be used when stating mandatory requirements; *will* is only used in statements of fact.

1.4 Normative References

The following standards contain provisions which, through references in POSIX.1-2024, constitute provisions of POSIX.1-2024. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on POSIX.1-2024 are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 646: 1991
: ISO/IEC 646: 1991, Information Processing — ISO 7-Bit Coded Character Set for Information Interchange.[3]

ISO 4217: 2015
: ISO 4217: 2015, Codes for the representation of currencies.

ISO 8601-1: 2019
: ISO 8601-1: 2019, Date and time — Representations for information interchange — Part 1: Basic rules.

ISO C (C17)
: ISO/IEC 9899: 2018, Programming Languages — C.

ISO/IEC 10646: 2020
: ISO/IEC 10646: 2020, Information Technology — Universal coded character set (UCS).

1.5 Change History

Change history is described in the Rationale (Informative) volume of POSIX.1-2024, and in the CHANGE HISTORY section of reference pages.

1.6 Terminology

For the purposes of POSIX.1-2024, the following terminology definitions apply:

can
: Describes a permissible optional feature or behavior available to the user or application. The feature or behavior is mandatory for an implementation that conforms to POSIX.1-2024. An application can rely on the existence of the feature or behavior.

implementation-defined
: Describes a value or behavior that is not defined by POSIX.1-2024 but is selected by an implementor. The value or behavior may vary among implementations that conform to POSIX.1-2024. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior cannot be assured to be portable across conforming implementations.

  The implementor shall document such a value or behavior so that it can be used correctly by an application.

legacy
: Describes a feature or behavior that is being retained for compatibility with older applications, but which has limitations which make it inappropriate for developing portable

3. ISO/IEC documents can be obtained from https://www.iso.org/store.html.

applications. New applications should use alternative means of obtaining equivalent functionality.

may
Describes a feature or behavior that is optional for an implementation that conforms to POSIX.1-2024. An application should not rely on the existence of the feature or behavior. An application that relies on such a feature or behavior cannot be assured to be portable across conforming implementations.

To avoid ambiguity, the opposite of *may* is expressed as *need not*, instead of *may not*.

shall
For an implementation that conforms to POSIX.1-2024, describes a feature or behavior that is mandatory. An application can rely on the existence of the feature or behavior.

For an application or user, describes a behavior that is mandatory.

should
For an implementation that conforms to POSIX.1-2024, describes a feature or behavior that is recommended but not mandatory. An application should not rely on the existence of the feature or behavior. An application that relies on such a feature or behavior cannot be assured to be portable across conforming implementations.

For an application, describes a feature or behavior that is recommended programming practice for optimum portability.

undefined
Describes the nature of a value or behavior not defined by POSIX.1-2024 which results from use of an invalid program construct or invalid data input.

The value or behavior may vary among implementations that conform to POSIX.1-2024. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

unspecified
Describes the nature of a value or behavior not specified by POSIX.1-2024 which results from use of a valid program construct or valid data input.

The value or behavior may vary among implementations that conform to POSIX.1-2024. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior cannot be assured to be portable across conforming implementations.

1.7 Definitions and Concepts

Definitions and concepts are defined in Chapter 3 (on page 31) and Chapter 4 (on page 95).

1.8 Portability

Some of the utilities in the Shell and Utilities volume of POSIX.1-2024 and functions in the System Interfaces volume of POSIX.1-2024 describe functionality that might not be fully portable to systems meeting the requirements for POSIX conformance (see Chapter 2, on page 15).

Where optional, enhanced, or reduced functionality is specified, the text is shaded and a code in the margin identifies the nature of the option, extension, or warning (see Section 1.8.1). For maximum portability, an application should avoid such functionality.

Unless the primary task of a utility is to produce textual material on its standard output, application developers should not rely on the format or content of any such material that may be produced. Where the primary task *is* to provide such material, but the output format is incompletely specified, the description is marked with the OF margin code and shading. Application developers are warned not to expect that the output of such an interface on one system is any guide to its behavior on another system.

1.8.1 Codes

The codes and their meanings are as follows. See also Section 1.8.2 (on page 12).

ADV Advisory Information
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the ADV margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the ADV margin legend.

CD C-Language Development Utilities
The functionality described is optional.

Where applicable, utilities are marked with the CD margin legend in the SYNOPSIS section. Where additional semantics apply to a utility, the material is identified by use of the CD margin legend.

CPT Process CPU-Time Clocks
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the CPT margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the CPT margin legend.

CX Extension to the ISO C standard
The functionality described is an extension to the ISO C standard or a deviation from it. Application developers can make use of the functionality as it is supported on all POSIX.1-2024-conforming systems.

With each function or header from the ISO C standard, a statement is included to the effect that ``any conflict is unintentional'', or ``any other conflict is unintentional'' if there is an intentional conflict (deviation). That is intended to refer to a direct conflict. POSIX.1-2024 acts in part as a profile of the ISO C standard, and it may choose to further constrain behaviors allowed to vary by the ISO C standard. Such limitations and other compatible differences are not considered conflicts, even if a CX mark is missing. The markings are for information only.

Where additional semantics apply to a function or header, the material is identified by use of the CX margin legend.

DC Device Control
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the DC margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the DC margin legend.

FR FORTRAN Runtime Utilities
The functionality described is optional.

Where applicable, utilities are marked with the FR margin legend in the SYNOPSIS section. Where additional semantics apply to a utility, the material is identified by use of the FR margin legend.

FSC File Synchronization
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the FSC margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the FSC margin legend.

IP6 IPV6
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the IP6 margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the IP6 margin legend.

MC1 Non-Robust Mutex Priority Protection or Non-Robust Mutex Priority Inheritance or Robust Mutex Priority Protection or Robust Mutex Priority Inheritance
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

This is a shorthand notation for combinations of multiple option codes.

Where applicable, functions are marked with the MC1 margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the MC1 margin legend.

Refer to Section 1.8.2 (on page 12).

ML Process Memory Locking
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the ML margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the ML margin legend.

MLR Range Memory Locking
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the MLR margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the MLR margin legend.

MSG **Message Passing**
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the MSG margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the MSG margin legend.

MX **IEC 60559 Floating-Point**
The functionality described is optional. The functionality described is mandated by the ISO C standard only for implementations that define __STDC_IEC_559__.

MXC **IEC 60559 Complex Floating-Point**
The functionality described is optional. The functionality described is mandated by the ISO C standard only for implementations that define __STDC_IEC_559_COMPLEX__.

MXX **IEC 60559 Floating-Point Extension**
The functionality described is optional. The functionality described is part of the IEC 60559 Floating-Point option, but is an extension to the ISO C standard.

OB **Obsolescent**
The functionality described may be removed in a future version of this volume of POSIX.1-2024. Strictly Conforming POSIX Applications and Strictly Conforming XSI Applications shall not use obsolescent features.

Where applicable, the material is identified by use of the OB margin legend.

OF **Output Format Incompletely Specified**
The functionality described is an XSI extension. The format of the output produced by the utility is not fully specified. It is therefore not possible to post-process this output in a consistent fashion. Typical problems include unknown length of strings and unspecified field delimiters.

Where applicable, the material is identified by use of the OF margin legend.

OH **Optional Header**
In the SYNOPSIS section of some interfaces in the System Interfaces volume of POSIX.1-2024 an included header is marked as in the following example:

OH
```
#include <sys/types.h>
#include <fcntl.h>
int open(const char *path, int oflag, ...);
```

The OH margin legend indicates that the optional header defines constants that will be needed if the function is called with certain flag arguments; thus it may be required for some of the functionality described, but is not needed otherwise.

PIO **Prioritized Input and Output**
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the PIO margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the PIO margin legend.

PS **Process Scheduling**
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the PS margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the PS

278 margin legend.

279 RPI Robust Mutex Priority Inheritance
280 The functionality described is optional. The functionality described is also an extension to the
281 ISO C standard.

282 Where applicable, functions are marked with the RPI margin legend in the SYNOPSIS section.
283 Where additional semantics apply to a function, the material is identified by use of the RPI
284 margin legend.

285 RPP Robust Mutex Priority Protection
286 The functionality described is optional. The functionality described is also an extension to the
287 ISO C standard.

288 Where applicable, functions are marked with the RPP margin legend in the SYNOPSIS section.
289 Where additional semantics apply to a function, the material is identified by use of the RPP
290 margin legend.

291 RS Raw Sockets
292 The functionality described is optional. The functionality described is also an extension to the
293 ISO C standard.

294 Where applicable, functions are marked with the RS margin legend in the SYNOPSIS section.
295 Where additional semantics apply to a function, the material is identified by use of the RS
296 margin legend.

297 SD Software Development Utilities
298 The functionality described is optional.

299 Where applicable, utilities are marked with the SD margin legend in the SYNOPSIS section.
300 Where additional semantics apply to a utility, the material is identified by use of the SD margin
301 legend.

302 SHM Shared Memory Objects
303 The functionality described is optional. The functionality described is also an extension to the
304 ISO C standard.

305 Where applicable, functions are marked with the SHM margin legend in the SYNOPSIS section.
306 Where additional semantics apply to a function, the material is identified by use of the SHM
307 margin legend.

308 SIO Synchronized Input and Output
309 The functionality described is optional. The functionality described is also an extension to the
310 ISO C standard.

311 Where applicable, functions are marked with the SIO margin legend in the SYNOPSIS section.
312 Where additional semantics apply to a function, the material is identified by use of the SIO
313 margin legend.

314 SPN Spawn
315 The functionality described is optional. The functionality described is also an extension to the
316 ISO C standard.

317 Where applicable, functions are marked with the SPN margin legend in the SYNOPSIS section.
318 Where additional semantics apply to a function, the material is identified by use of the SPN
319 margin legend.

320 SS Process Sporadic Server
321 The functionality described is optional. The functionality described is also an extension to the
322 ISO C standard.

Where applicable, functions are marked with the SS margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the SS margin legend.

TCT Thread CPU-Time Clocks

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TCT margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TCT margin legend.

TPI Non-Robust Mutex Priority Inheritance

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TPI margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TPI margin legend.

TPP Non-Robust Mutex Priority Protection

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TPP margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TPP margin legend.

TPS Thread Execution Scheduling

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TPS margin legend for the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TPS margin legend.

TSA Thread Stack Address Attribute

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TSA margin legend for the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TSA margin legend.

TSH Thread Process-Shared Synchronization

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TSH margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TSH margin legend.

TSP Thread Sporadic Server

The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TSP margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TSP margin legend.

TSS Thread Stack Size Attribute
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TSS margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TSS margin legend.

TYM Typed Memory Objects
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the TYM margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the TYM margin legend.

UP User Portability Utilities
The functionality described is optional.

Where applicable, utilities are marked with the UP margin legend in the SYNOPSIS section. Where additional semantics apply to a utility, the material is identified by use of the UP margin legend.

UU UUCP Utilities
The functionality described is optional. The functionality described is also an extension to the ISO C standard.

Where applicable, functions are marked with the UU margin legend in the SYNOPSIS section. Where additional semantics apply to a function, the material is identified by use of the UU margin legend.

XSI X/Open System Interfaces
The functionality described is part of the X/Open Systems Interfaces option. Functionality marked XSI is an extension to the ISO C standard. Application developers may confidently make use of such extensions on all systems supporting the X/Open System Interfaces option.

If an entire SYNOPSIS section is shaded and marked XSI, all the functionality described in that reference page is an extension. See Section 2.1.4 (on page 19).

1.8.2 Margin Code Notation

Some of the functionality described in POSIX.1-2024 depends on support of more than one option, or independently may depend on several options. The following notation for margin codes is used to denote the following cases.

A Feature Dependent on One or Two Options

In this case, margin codes have a <space> separator; for example:

SHM This feature requires support for only the Shared Memory Objects option.

SHM TYM This feature requires support for both the Shared Memory Objects option and the Typed Memory Objects option; that is, an application which uses this feature is portable only between implementations that provide both options.

A Feature Dependent on Either of the Options Denoted

In this case, margin codes have a '|' separator to denote the logical OR; for example:

SHM|TYM This feature is dependent on support for either the Shared Memory Objects option or the Typed Memory Objects option; that is, an application which uses this feature is portable between implementations that provide any (or all) of the options.

A Feature Dependent on More than Two Options

The following shorthand notations are used:

MC1 The MC1 margin code is shorthand for TPP|TPI|RPP|RPI. Features which are shaded with this margin code require support of either the Non-Robust Mutex Priority Protection option or the Non-Robust Mutex Priority Inheritance option or the Robust Mutex Priority Protection option or the Robust Mutex Priority Inheritance option.

Large Sections Dependent on an Option

Where large sections of text are dependent on support for an option, a lead-in text block is provided and shaded accordingly; for example:

XSI This section describes extensions to support interprocess communication. The functionality described in this section shall be provided on implementations that support the XSI option (and the rest of this section is not further shaded).

Chapter 2

Conformance

2.1 Implementation Conformance

For the purposes of POSIX.1-2024, the implementation conformance requirements given in this section apply.

2.1.1 Requirements

A *conforming implementation* shall meet all of the following criteria:

1. The system shall support all utilities, functions, and facilities defined within POSIX.1-2024 that are required for POSIX conformance (see Section 2.1.3, on page 17). These interfaces shall support the functional behavior described herein.
2. The system may support the X/Open System Interfaces (XSI) option as described in Section 2.1.4 (on page 19).
3. The system may support one or more options as described under Section 2.1.5 (on page 20). When an implementation claims that an option is supported, all of its constituent parts shall be provided.
4. The system may provide non-standard extensions. These are features not required by POSIX.1-2024 and may include, but are not limited to:
   - Additional functions
   - Additional headers
   - Additional symbols in standard headers
   - Additional utilities
   - Additional options for standard utilities
   - Additional environment variables
   - Additional file types
   - Non-conforming file systems (for example, legacy file systems for which _POSIX_NO_TRUNC is false, case-insensitive file systems, or network file systems)
   - Dynamically populated file systems (for example, **/proc**)
   - Additional character special files with special properties (for example, **/dev/stdin**, **/dev/stdout**, and **/dev/stderr**)

   Non-standard extensions of the utilities, functions, or facilities specified in POSIX.1-2024 should be identified as such in the system documentation. Non-standard extensions, when used, may change the behavior of utilities, functions, or facilities defined by POSIX.1-2024. The conformance document shall define an environment in which an application can be run with the behavior specified by POSIX.1-2024. In no case shall such an environment require modification of a Strictly Conforming POSIX Application (see

Section 2.2.1, on page 27).

Note: If the documented method of setting up a conforming environment includes the need to set one or more environment variables, then the values of those environment variables cannot include any <space> characters, since the *confstr*( ) function has to be able to return them in a <space>-separated list of variable=value pairs. See XSH *confstr*( ) (on page 763).

2.1.2 Documentation

A conformance document with the following information shall be available for an implementation claiming conformance to POSIX.1-2024. The conformance document shall have the same structure as POSIX.1-2024, with the information presented in the appropriate sections and subsections. Sections and subsections that consist solely of subordinate section titles, with no other information, are not required. The conformance document shall not contain information about extended facilities or capabilities outside the scope of POSIX.1-2024.

The conformance document shall contain a statement that indicates the full name, number, and date of the standard that applies. The conformance document may also list international software standards that are available for use by a Conforming POSIX Application. Applicable characteristics where documentation is required by one of these standards, or by standards of government bodies, may also be included.

The conformance document shall describe the limit values found in the headers **<limits.h>** (on page 282) and **<unistd.h>** (on page 458), stating values, the conditions under which those values may change, and the limits of such variations, if any.

The conformance document shall describe the behavior of the implementation for all implementation-defined features defined in POSIX.1-2024. This requirement shall be met by listing these features and providing either a specific reference to the system documentation or providing full syntax and semantics of these features. When the value or behavior in the implementation is designed to be variable or customized on each instantiation of the system, the implementation provider shall document the nature and permissible ranges of this variation.

The conformance document may specify the behavior of the implementation for those features where POSIX.1-2024 states that implementations may vary or where features are identified as undefined or unspecified.

The conformance document shall not contain documentation other than that specified in the preceding paragraphs except where such documentation is specifically allowed or required by other provisions of POSIX.1-2024.

The phrases ``shall document'' or ``shall be documented'' in POSIX.1-2024 mean that documentation of the feature shall appear in the conformance document, as described previously, unless there is an explicit reference in the conformance document to show where the information can be found in the system documentation.

The system documentation should also contain the information found in the conformance document.

2.1.3 POSIX Conformance

A conforming implementation shall meet the following criteria for POSIX conformance.

2.1.3.1 POSIX System Interfaces

The following requirements apply to the system interfaces (functions and headers):

- The system shall support all the mandatory functions and headers defined in POSIX.1-2024, and shall set the symbolic constant _POSIX_VERSION to the value 202405L.
- Although all implementations conforming to POSIX.1-2024 support all the features described below, there may be system-dependent or file system-dependent configuration procedures that can remove or modify any or all of these features. Such configurations should not be made if strict compliance is required.

  The following symbolic constants shall be defined with a value other than –1. If a constant is defined with the value zero, applications should use the *sysconf*( ), *pathconf*( ), or *fpathconf*( ) functions, or the *getconf* utility, to determine which features are present on the system at that time or for the particular pathname in question.

  — _POSIX_CHOWN_RESTRICTED

    The use of *chown*( ) is restricted to a process with appropriate privileges, and to changing the group ID of a file only to the effective group ID of the process or to one of its supplementary group IDs.

  — _POSIX_NO_TRUNC

    Pathname components longer than {NAME_MAX} generate an error.

- The following symbolic constants shall be defined by the implementation as follows:

  — Symbolic constants defined with the value 202405L:

    _POSIX_ASYNCHRONOUS_IO
    _POSIX_BARRIERS
    _POSIX_CLOCK_SELECTION
    _POSIX_MAPPED_FILES
    _POSIX_MEMORY_PROTECTION
    _POSIX_MONOTONIC_CLOCK
    _POSIX_READER_WRITER_LOCKS
    _POSIX_REALTIME_SIGNALS
    _POSIX_SEMAPHORES
    _POSIX_SPIN_LOCKS
    _POSIX_THREAD_SAFE_FUNCTIONS
    _POSIX_THREADS
    _POSIX_TIMEOUTS
    _POSIX_TIMERS
    _POSIX2_C_BIND

  — Symbolic constants defined with a value greater than zero:

    _POSIX_JOB_CONTROL
    _POSIX_REGEXP
    _POSIX_SAVED_IDS
    _POSIX_SHELL

— Symbolic constants defined with a value other than –1.

_POSIX_VDISABLE

Note: The symbols above represent historical options that are no longer allowed as options, but are retained here for backwards-compatibility of applications.

- The system may support one or more options (see Section 2.1.6, on page 25) denoted by the following symbolic constants:

_POSIX_ADVISORY_INFO
_POSIX_CPUTIME
_POSIX_DEVICE_CONTROL
_POSIX_FSYNC
_POSIX_IPV6
_POSIX_MEMLOCK
_POSIX_MEMLOCK_RANGE
_POSIX_MESSAGE_PASSING
_POSIX_PRIORITIZED_IO
_POSIX_PRIORITY_SCHEDULING
_POSIX_RAW_SOCKETS
_POSIX_SHARED_MEMORY_OBJECTS
_POSIX_SPAWN
_POSIX_SPORADIC_SERVER
_POSIX_SYNCHRONIZED_IO
_POSIX_THREAD_ATTR_STACKADDR
_POSIX_THREAD_CPUTIME
_POSIX_THREAD_ATTR_STACKSIZE
_POSIX_THREAD_PRIO_INHERIT
_POSIX_THREAD_PRIO_PROTECT
_POSIX_THREAD_PRIORITY_SCHEDULING
_POSIX_THREAD_PROCESS_SHARED
_POSIX_THREAD_SPORADIC_SERVER
_POSIX_TYPED_MEMORY_OBJECTS
_XOPEN_CRYPT
_XOPEN_REALTIME
_XOPEN_REALTIME_THREADS
_XOPEN_UNIX

If the Advisory Information option is supported, there shall be at least one file system that supports the functionality.

2.1.3.2 *POSIX Shell and Utilities*

The following requirements apply to the shell and utilities:

- The system shall provide all the mandatory utilities in the Shell and Utilities volume of POSIX.1-2024 with all the functional behavior described therein.
- The system shall support the Large File capabilities described in the Shell and Utilities volume of POSIX.1-2024.
- The system may support one or more options (see Section 2.1.6, on page 25) denoted by the following symbolic constants. (The literal names below apply to the *getconf* utility.)

POSIX2_C_DEV
POSIX2_CHAR_TERM
POSIX2_FORT_RUN
POSIX2_LOCALEDEF
POSIX2_SW_DEV
POSIX2_UPE
XOPEN_UNIX
XOPEN_UUCP

Additional language bindings and development utility options may be provided in other related standards or in a future version of this standard. In the former case, additional symbolic constants of the same general form as shown in this subsection should be defined by the related standard document and made available to the application without requiring POSIX.1-2024 to be updated.

2.1.4 XSI Conformance

XSI This section describes the criteria for implementations providing conformance to the X/Open System Interfaces (XSI) option (see Section 3.424, on page 93). The functionality described in this section shall be provided on implementations that support the XSI option (and the rest of this section is not further shaded).

POSIX.1-2024 describes utilities, functions, and facilities offered to application programs by the X/Open System Interfaces (XSI) option. An XSI-conforming implementation shall meet the criteria for POSIX conformance and the following requirements listed in this section.

XSI-conforming implementations shall set the symbolic constant _XOPEN_UNIX to a value other than –1 and shall set the symbolic constant _XOPEN_VERSION to the value 800.

2.1.4.1 *XSI System Interfaces*

The following requirements apply to the system interfaces when the XSI option is supported:

- The system shall support all the functions and headers defined in POSIX.1-2024 as part of the XSI option denoted by the XSI marking in the SYNOPSIS section, and any extensions marked with the XSI option marking (see Section 1.8.1, on page 7) within the text.
- The system shall support the following options defined within POSIX.1-2024 (see Section 2.1.6, on page 25):

  _POSIX_FSYNC
  _POSIX_THREAD_ATTR_STACKADDR
  _POSIX_THREAD_ATTR_STACKSIZE
  _POSIX_THREAD_PROCESS_SHARED

- The system may support the following XSI Option Groups (see Section 2.1.5.2, on page 22) defined within POSIX.1-2024:
  - — Encryption
  - — Realtime
  - — Advanced Realtime

— Realtime Threads

— Advanced Realtime Threads

2.1.4.2 *XSI Shell and Utilities Conformance*

The following requirements apply to the shell and utilities when the XSI option is supported:

- The system shall support all the utilities defined in the Shell and Utilities volume of POSIX.1-2024 as part of the XSI option denoted by the XSI marking in the SYNOPSIS section, and any extensions marked with the XSI option marking (see Section 1.8.1, on page 7) within the text.
- The system shall support the User Portability Utilities option and the Terminal Characteristics option.
- The system shall support creation of locales (see Chapter 7, on page 127).
- The C-language Development utility *c17* shall be supported.
- The XSI Development Utilities option may be supported. It consists of the following software development utilities:

admin	*delta*	*rmdel*	*val*
cflow	*get*	*sact*	*what*
ctags	*nm*	*sccs*	
cxref	*prs*	*unget*	

2.1.5 Option Groups

An Option Group is a group of related functions or options defined within the System Interfaces volume of POSIX.1-2024.

If an implementation supports an Option Group, then the system shall support the functional behavior described herein.

If an implementation does not support an Option Group, then the system need not support the functional behavior described herein.

2.1.5.1 *Subprofiling Considerations*

Profiling standards supporting functional requirements less than that required in POSIX.1-2024 may subset both mandatory and optional functionality required for POSIX Conformance (see Section 2.1.3, on page 17) or XSI Conformance (see Section 2.1.4, on page 19). Such profiles shall organize the subsets into Subprofiling Option Groups.

XRAT Appendix E (on page 3943) describes a representative set of such Subprofiling Option Groups for use by profiles applicable to specialized realtime systems. POSIX.1-2024 does not require that the presence of Subprofiling Option Groups be testable at compile-time (as symbols defined in any header) or at runtime (via *sysconf*( ) or *getconf*).

A Subprofiling Option Group may provide basic system functionality that other Subprofiling Option Groups and other options depend upon.[4] If a profile of POSIX.1-2024 does not require an implementation to provide a Subprofiling Option Group that provides features utilized by a required Subprofiling Option Group (or option),[5] the profile shall specify[6] all of the following:

- Restricted or altered behavior of interfaces defined in POSIX.1-2024 that may differ on an implementation of the profile
- Additional behaviors that may produce undefined or unspecified results
- Additional implementation-defined behavior that implementations shall be required to document in the profile's conformance document

if any of the above is a result of the profile not requiring an interface required by POSIX.1-2024.

The following additional rules shall apply to all profiles of POSIX.1-2024:

- Any application that conforms to that profile shall also conform to POSIX.1-2024, unless the application depends on the definition of a profile support indicator macro in **<unistd.h>** (that is, a profile shall not require restricted, altered, or extended behaviors of an implementation of POSIX.1-2024).
- Profiles are permitted to require the definition of a *profile support indicator macro* with a name beginning _POSIX_AEP_ in **<unistd.h>**.
- Profiles shall require the definition of the macro _POSIX_SUBPROFILE in **<unistd.h>** on implementations that do not meet all of the requirements of a POSIX.1-conforming implementation.
- Profiles are permitted to add additional requirements to the limits defined in **<limits.h>** and **<stdint.h>**, subject to the following:

  For the limits in **<limits.h>** and **<stdint.h>**:

  — If the limit is specified as having a fixed value, it shall not be changed by a profile.

  — If a limit is specified as having a minimum or maximum acceptable value, it may be changed by a profile as follows:

    — A profile may increase a minimum acceptable value, but shall not make a minimum acceptable value smaller.

    — A profile may reduce a maximum acceptable value, but shall not make a maximum acceptable value larger.
- A profile shall not change a limit specified as having a minimum or maximum value into a limit specified as having a fixed value.
- A profile shall not create new limits.
- Any implementation that conforms to POSIX.1-2024 (including all options and extended limits required by the profile) shall also conform to that profile, except for the possible omission from **<unistd.h>** of a profile support indicator macro required by the profile.

---

4. As an example, the File System profiling option group provides underlying support for pathname resolution and file creation which are needed by any interface in POSIX.1-2024 that parses a *path* argument. If a profile requires support for the Device Input and Output profiling option group but does not require support for the File System profiling option group, the profile needs to specify how pathname resolution is to behave in that profile, how the O_CREAT flag to *open*( ) is to be handled (and the use of the character `'a'` in the *mode* argument of *fopen*( ) when a pathname argument names a file that does not exist), and specify lots of other details.

5. As an example, POSIX.1-2024 requires that implementations claiming to support the Range Memory Locking option also support the Process Memory Locking option. A profile could require that the Range Memory Locking option had to be supplied without requiring that the Process Memory Locking option be supplied as long as the profile specifies everything an application developer or system implementor would have to know to build an application or implementation conforming to the profile.

6. Note that the profile could just specify that any use of the features not specified by the profile would produce undefined or unspecified results.

2.1.5.2 *XSI Option Groups*

XSI This section describes Option Groups to support the definition of XSI conformance within the System Interfaces volume of POSIX.1-2024. The functionality described in this section shall be provided on implementations that support the XSI option and the appropriate Option Group (and the rest of this section is not further shaded).

The following Option Groups are defined.

Encryption

The Encryption Option Group is denoted by the symbolic constant _XOPEN_CRYPT. It includes the following functions:

OB *crypt*( ), *encrypt*( ), *setkey*( )

These functions are marked CRYPT.

Due to export restrictions on the cryptographic algorithm in some countries, implementations may be restricted in making these functions available. All the functions in the Encryption Option Group may therefore return [ENOSYS] or, alternatively, *encrypt*( ) shall return [ENOSYS] for the decryption operation.

An implementation that claims conformance to this Option Group shall set _XOPEN_CRYPT to a value other than –1.

Realtime

The Realtime Option Group is denoted by the symbolic constant _XOPEN_REALTIME.

This Option Group includes a set of realtime functions drawn from options within POSIX.1-2024 (see Section 2.1.6, on page 25).

Where entire functions are included in the Option Group, the NAME section is marked with REALTIME. Where additional semantics have been added to existing pages, the new material is identified by use of the appropriate margin legend for the underlying option defined within POSIX.1-2024.

An implementation that claims conformance to this Option Group shall set _XOPEN_REALTIME to a value other than –1.

This Option Group consists of the set of the following options from within POSIX.1-2024 (see Section 2.1.6, on page 25):

_POSIX_FSYNC
_POSIX_MEMLOCK
_POSIX_MEMLOCK_RANGE
_POSIX_MESSAGE_PASSING
_POSIX_PRIORITIZED_IO
_POSIX_PRIORITY_SCHEDULING
_POSIX_SHARED_MEMORY_OBJECTS
_POSIX_SYNCHRONIZED_IO

If the symbolic constant _XOPEN_REALTIME is defined to have a value other than –1, then the following symbolic constants shall be defined by the implementation to have the value 202405L:

742 _POSIX_MEMLOCK
743 _POSIX_MEMLOCK_RANGE
744 _POSIX_MESSAGE_PASSING
745 _POSIX_PRIORITY_SCHEDULING
746 _POSIX_SHARED_MEMORY_OBJECTS
747 _POSIX_SYNCHRONIZED_IO

748 The functionality associated with _POSIX_FSYNC shall always be supported on XSI-conformant
749 systems.

750 Support of _POSIX_PRIORITIZED_IO on XSI-conformant systems is optional. If
751 _POSIX_PRIORITIZED_IO is supported, then asynchronous I/O operations performed by
752 *aio_read*(), *aio_write*(), and *lio_listio*() shall be submitted at a priority equal to the scheduling
753 priority equal to a base scheduling priority minus *aiocbp*−>*aio_reqprio*. If Thread Execution
754 Scheduling is not supported, then the base scheduling priority is that of the calling process;
755 otherwise, the base scheduling priority is that of the calling thread. The implementation shall
756 also document for which files I/O prioritization is supported.

757 **Advanced Realtime**

758 An implementation that claims conformance to this Option Group shall also support the
759 Realtime Option Group.

760 Where entire functions are included in the Option Group, the NAME section is marked with
761 ADVANCED REALTIME. Where additional semantics have been added to existing pages, the
762 new material is identified by use of the appropriate margin legend for the underlying option
763 defined within POSIX.1-2024.

764 This Option Group consists of the set of the following options from within POSIX.1-2024 (see
765 Section 2.1.6, on page 25):

766 _POSIX_ADVISORY_INFO
767 _POSIX_CPUTIME
768 _POSIX_SPAWN
769 _POSIX_SPORADIC_SERVER
770 _POSIX_TYPED_MEMORY_OBJECTS

771 If the implementation supports the Advanced Realtime Option Group, then the following
772 symbolic constants shall be defined by the implementation to have the value 202405L:

773 _POSIX_ADVISORY_INFO
774 _POSIX_CPUTIME
775 _POSIX_SPAWN
776 _POSIX_SPORADIC_SERVER
777 _POSIX_TYPED_MEMORY_OBJECTS

778 If the symbolic constant _POSIX_SPORADIC_SERVER is defined, then the symbolic constant
779 _POSIX_PRIORITY_SCHEDULING shall also be defined by the implementation to have the
780 value 202405L.

Realtime Threads

The Realtime Threads Option Group is denoted by the symbolic constant _XOPEN_REALTIME_THREADS.

This Option Group consists of the set of the following options from within POSIX.1-2024 (see Section 2.1.6, on page 25):

_POSIX_THREAD_PRIO_INHERIT
_POSIX_THREAD_PRIO_PROTECT
_POSIX_THREAD_PRIORITY_SCHEDULING
_POSIX_THREAD_ROBUST_PRIO_INHERIT
_POSIX_THREAD_ROBUST_PRIO_PROTECT

Where applicable, whole pages are marked REALTIME THREADS, together with the appropriate option margin legend for the SYNOPSIS section (see Section 1.8.1, on page 7).

An implementation that claims conformance to this Option Group shall set _XOPEN_REALTIME_THREADS to a value other than –1.

If the symbol _XOPEN_REALTIME_THREADS is defined to have a value other than –1, then the following options shall also be defined by the implementation to have the value 202405L:

_POSIX_THREAD_PRIO_INHERIT
_POSIX_THREAD_PRIO_PROTECT
_POSIX_THREAD_PRIORITY_SCHEDULING
_POSIX_THREAD_ROBUST_PRIO_INHERIT
_POSIX_THREAD_ROBUST_PRIO_PROTECT

Advanced Realtime Threads

An implementation that claims conformance to this Option Group shall also support the Realtime Threads Option Group.

Where entire functions are included in the Option Group, the NAME section is marked with ADVANCED REALTIME THREADS. Where additional semantics have been added to existing pages, the new material is identified by use of the appropriate margin legend for the underlying option defined within POSIX.1-2024.

This Option Group consists of the set of the following options from within POSIX.1-2024 (see Section 2.1.6, on page 25):

_POSIX_THREAD_CPUTIME
_POSIX_THREAD_SPORADIC_SERVER

If the symbolic constant _POSIX_THREAD_SPORADIC_SERVER is defined to have the value 202405L, then the symbolic constant _POSIX_THREAD_PRIORITY_SCHEDULING shall also be defined by the implementation to have the value 202405L.

If the implementation supports the Advanced Realtime Threads Option Group, then the following symbolic constants shall be defined by the implementation to have the value 202405L:

_POSIX_THREAD_CPUTIME
_POSIX_THREAD_SPORADIC_SERVER

2.1.6 Options

The symbolic constants defined in **<unistd.h>**, Constants for Options and Option Groups (on page 458) reflect implementation options for POSIX.1-2024. These symbols can be used by the application to determine which of three categories of support for optional facilities are provided by the implementation.

1. Option not supported for compilation.

   The implementation advertises at compile time (by defining the constant in **<unistd.h>** with value –1, or by leaving it undefined) that the option is not supported for compilation and, at the time of compilation, is not supported for runtime use. In this case, the headers, data types, function interfaces, and utilities required only for the option need not be present. A later runtime check using the *fpathconf*( ), *pathconf*( ), or sysconf functions defined in the System Interfaces volume of POSIX.1-2024 or the *getconf* utility defined in the Shell and Utilities volume of POSIX.1-2024 can in some circumstances indicate that the option is supported at runtime. (For example, an old application binary might be run on a newer implementation to which support for the option has been added.)

2. Option always supported.

   The implementation advertises at compile time (by defining the constant in **<unistd.h>** with a value greater than zero) that the option is supported both for compilation and for use at runtime. In this case, all headers, data types, function interfaces, and utilities required only for the option shall be available and shall operate as specified. Runtime checks with *fpathconf*( ), *pathconf*( ), or sysconf shall indicate that the option is supported.

3. Option might or might not be supported at runtime.

   The implementation advertises at compile time (by defining the constant in **<unistd.h>** with value zero) that the option is supported for compilation and might or might not be supported at runtime. In this case, the *fpathconf*( ), *pathconf*( ), or *sysconf*( ) functions defined in the System Interfaces volume of POSIX.1-2024 or the *getconf* utility defined in the Shell and Utilities volume of POSIX.1-2024 can be used to retrieve the value of each symbol on each specific implementation to determine whether the option is supported at runtime. All headers, data types, and function interfaces required to compile and execute applications which use the option at runtime (after checking at runtime that the option is supported) shall be provided, but if the option is not supported at runtime they need not operate as specified. Utilities or other facilities required only for the option, but not needed to compile and execute such applications, need not be present.

If an option is not supported for compilation, an application that attempts to use anything associated only with the option is considered to be requiring an extension. Unless explicitly specified otherwise, the behavior of functions associated with an option that is not supported at runtime is unspecified, and an application that uses such functions without first checking *fpathconf*( ), *pathconf*( ), or sysconf is considered to be requiring an extension.

Margin codes are defined for each option (see Section 1.8.1, on page 7).

2.1.6.1 System Interfaces

Refer to **<unistd.h>**, Constants for Options and Option Groups (on page 458) for the list of options.

862 *2.1.6.2 Shell and Utilities*

863 Each of these symbols shall be considered valid names by the implementation. Refer to
864 **<unistd.h>**, Constants for Options and Option Groups (on page 458).

865 The literal names shown below apply only to the *getconf* utility.

866 CD POSIX2_C_DEV
867 The system supports the C-Language Development Utilities option.

868 The utilities in the C-Language Development Utilities option are used for the development
869 of C-language applications, including compilation or translation of C source code and
870 complex program generators for simple lexical tasks and processing of context-free
871 grammars.

872 The utilities listed below may be provided by a conforming system; however, any system
873 claiming conformance to the C-Language Development Utilities option shall provide all of
874 the utilities listed.

875 *c17*
876 *lex*
877 *yacc*

878 POSIX2_CHAR_TERM
879 The system supports the Terminal Characteristics option. This value need not be present on
880 a system not supporting the User Portability Utilities option.

881 Where applicable, the dependency is noted within the description of the utility.

882 This option applies only to systems supporting the User Portability Utilities option. If
883 supported, then the system supports at least one terminal type capable of all operations
884 described in POSIX.1-2024; see Section 10.2 (on page 197).

885 FR POSIX2_FORT_RUN
886 The system supports the FORTRAN Runtime Utilities option.

887 The *asa* utility is the only utility in the FORTRAN Runtime Utilities option.

888 The *asa* utility may be provided by a conforming system; however, any system claiming
889 conformance to the FORTRAN Runtime Utilities option shall provide the *asa* utility.

890 POSIX2_LOCALEDEF
891 The system supports the Locale Creation Utilities option.

892 If supported, the system supports the creation of locales as described in the *localedef* utility.

893 The *localedef* utility may be provided by a conforming system; however, any system
894 claiming conformance to the Locale Creation Utilities option shall provide the *localedef*
895 utility.

896 SD POSIX2_SW_DEV
897 The system supports the Software Development Utilities option.

898 The utilities in the Software Development Utilities option are used for the development of
899 applications, including compilation or translation of source code, the creation and
900 maintenance of library archives, and the maintenance of groups of inter-dependent
901 programs.

902 The utilities listed below may be provided by the conforming system; however, any system
903 claiming conformance to the Software Development Utilities option shall provide all of the
904 utilities listed here.

ar
make
nm
strip

UP POSIX2_UPE

The system supports the User Portability Utilities option.

The utilities in the User Portability Utilities option shall be implemented on all systems that claim conformance to this option, except for the *vi* utility which is noted as having features that cannot be implemented on all terminal types; if the POSIX2_CHAR_TERM option is supported, the system shall support all such features on at least one terminal type; see Section 10.2 (on page 197).

The list of utilities in the User Portability Utilities option is as follows:

bg *fc* *jobs* *more* *vi*
ex *fg* *man* *talk*

XSI XOPEN_UNIX

The system supports the X/Open System Interfaces (XSI) option (see Section 2.1.4, on page 19).

UU XOPEN_UUCP

The system supports the UUCP Utilities option.

The list of utilities in the UUCP Utilities option is as follows:

uucp
uustat
uux

2.2 Application Conformance

For the purposes of POSIX.1-2024, the application conformance requirements given in this section apply.

All applications claiming conformance to POSIX.1-2024 shall use only language-dependent services for the C programming language described in Section 2.3 (on page 29), shall use only the utilities and facilities defined in the Shell and Utilities volume of POSIX.1-2024, and shall fall within one of the following categories.

2.2.1 Strictly Conforming POSIX Application

A Strictly Conforming POSIX Application is an application that requires only the facilities described in POSIX.1-2024. Such an application:

1. Shall accept any implementation behavior that results from actions it takes in areas described in POSIX.1-2024 as *implementation-defined* or *unspecified*, or where POSIX.1-2024 indicates that implementations may vary
2. Shall not perform any actions that are described as producing *undefined* results
3. For symbolic constants, shall accept any value in the range permitted by POSIX.1-2024, but shall not rely on any value in the range being greater than the minimums listed or being less than the maximums listed in POSIX.1-2024

4. Shall not use facilities designated as *obsolescent*
5. Is required to tolerate and permitted to adapt to the presence or absence of optional facilities whose availability is indicated by Section 2.1.3 (on page 17)
6. For the C programming language, shall not produce any output dependent on any behavior described in the ISO C standard as *unspecified*, *undefined*, or *implementation-defined*, unless the System Interfaces volume of POSIX.1-2024 specifies the behavior
7. For the C programming language, shall not exceed any minimum implementation limit defined in the ISO C standard, unless the System Interfaces volume of POSIX.1-2024 specifies a higher minimum implementation limit
8. For the C programming language, shall define _POSIX_C_SOURCE to be 202405L before any header is included

Within POSIX.1-2024, any restrictions placed upon a Conforming POSIX Application shall restrict a Strictly Conforming POSIX Application.

2.2.2 Conforming POSIX Application

2.2.2.1 ISO/IEC Conforming POSIX Application

An ISO/IEC Conforming POSIX Application is an application that uses only the facilities described in POSIX.1-2024 and approved Conforming Language bindings for any ISO or IEC standard. Such an application shall include a statement of conformance that documents all options and limit dependencies, and all other ISO or IEC standards used.

2.2.2.2 <National Body> Conforming POSIX Application

A <National Body> Conforming POSIX Application differs from an ISO/IEC Conforming POSIX Application in that it also may use specific standards of a single ISO/IEC member body referred to here as <*National Body*>. Such an application shall include a statement of conformance that documents all options and limit dependencies, and all other <National Body> standards used.

2.2.3 Conforming POSIX Application Using Extensions

A Conforming POSIX Application Using Extensions is an application that differs from a Conforming POSIX Application only in that it uses non-standard facilities that are consistent with POSIX.1-2024. Such an application shall fully document its requirements for these extended facilities, in addition to the documentation required of a Conforming POSIX Application. A Conforming POSIX Application Using Extensions shall be either an ISO/IEC Conforming POSIX Application Using Extensions or a <National Body> Conforming POSIX Application Using Extensions (see Section 2.2.2.1 and Section 2.2.2.2).

2.2.4 Strictly Conforming XSI Application

A Strictly Conforming XSI Application is an application that requires only the facilities described in POSIX.1-2024. Such an application:

1. Shall accept any implementation behavior that results from actions it takes in areas described in POSIX.1-2024 as *implementation-defined* or *unspecified*, or where POSIX.1-2024 indicates that implementations may vary
2. Shall not perform any actions that are described as producing *undefined* results
3. For symbolic constants, shall accept any value in the range permitted by POSIX.1-2024, but shall not rely on any value in the range being greater than the minimums listed or being less than the maximums listed in POSIX.1-2024
4. Shall not use facilities designated as *obsolescent*
5. Is required to tolerate and permitted to adapt to the presence or absence of optional facilities whose availability is indicated by Section 2.1.4 (on page 19)
6. For the C programming language, shall not produce any output dependent on any behavior described in the ISO C standard as *unspecified*, *undefined*, or *implementation-defined*, unless the System Interfaces volume of POSIX.1-2024 specifies the behavior
7. For the C programming language, shall not exceed any minimum implementation limit defined in the ISO C standard, unless the System Interfaces volume of POSIX.1-2024 specifies a higher minimum implementation limit
8. For the C programming language, shall define _XOPEN_SOURCE to be 800 before any header is included

Within POSIX.1-2024, any restrictions placed upon a Conforming POSIX Application shall restrict a Strictly Conforming XSI Application.

2.2.5 Conforming XSI Application Using Extensions

A Conforming XSI Application Using Extensions is an application that differs from a Strictly Conforming XSI Application only in that it uses non-standard facilities that are consistent with POSIX.1-2024. Such an application shall fully document its requirements for these extended facilities, in addition to the documentation required of a Strictly Conforming XSI Application.

2.3 Language-Dependent Services for the C Programming Language

Implementors seeking to claim conformance using the ISO C standard shall claim POSIX conformance as described in Section 2.1.3 (on page 17).

2.4 Other Language-Related Specifications

POSIX.1-2024 is currently specified in terms of the shell command language and ISO C. Bindings to other programming languages are being developed.

If conformance to POSIX.1-2024 is claimed for implementation of any programming language, the implementation of that language shall support the use of external symbols distinct to at least 31 bytes in length in the source program text. (That is, identifiers that differ at or before the thirty-first byte shall be distinct.) If a national or international standard governing a language defines a maximum length that is less than this value, the language-defined maximum shall be

supported. External symbols that differ only by case shall be distinct when the character set in use distinguishes uppercase and lowercase characters and the language permits (or requires) uppercase and lowercase characters to be distinct in external symbols.

