



**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**

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.

ISO/IEC/IEEE 9945:2026(en)

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

ISO/IEC/IEEE 9945:2026(en)

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

Copyright © 2024 by The Institute of Electrical and Electronics Engineers, Inc. and The Open Group
All rights reserved.

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

IEEE is a registered trademark in the U.S. Patent & Trademark Office and POSIX is a trademark owned by The Institute of Electrical and Electronics Engineers, Incorporated.

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.

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <https://www.ieee.org/about/corporate/governance/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher. Permission to reproduce all or any part of this standard must be with the consent of both copyright holders and may be subject to a license fee. Both copyright holders will need to be satisfied that the other has granted permission. Requests should be sent by email to austin-group-permissions@opengroup.org.

ISO/IEC/IEEE 9945:2026(en)

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

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>.

ISO/IEC/IEEE 9945:2026(en)

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

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.

ISO/IEC/IEEE 9945:2026(en)

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

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.

ISO/IEC/IEEE 9945:2026(en)

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

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

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>.

ISO/IEC/IEEE 9945:2026(en)

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

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>.

ISO/IEC/IEEE 9945:2026(en)

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

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

ISO/IEC/IEEE 9945:2026(en)

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

Austin Group Working Group Members

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

ISO/IEC/IEEE 9945:2026(en)

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

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

ISO/IEC/IEEE 9945:2026(en)

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

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.

ISO/IEC/IEEE 9945:2026(en)

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

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

ISO/IEC/IEEE 9945:2026(en)

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

- 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:

ISO/IEC/IEEE 9945:2026(en)

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

- 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 *regex()* 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:

ISO/IEC/IEEE 9945:2026(en)

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

- 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	

ISO/IEC/IEEE 9945:2026(en)

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

Reference	Example	Notes
C-Language Data Structure Member	<i>aio_lio_opcode</i>	
C-Language Data Type	long	
C-Language External Variable	<i>errno</i>	
C-Language Function	<i>system()</i>	
C-Language Function Argument	<i>arg</i>	
C-Language Function Family	<i>exec</i>	
C-Language Header	<sys/stat.h>	
C-Language Keyword	return	
C-Language Macro with Argument	<i>assert()</i>	
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	<i>PATH</i>	
Error Number	[EINTR]	
Example Output	Hello, World	
Filename	<i>/tmp</i>	
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	<i>awk</i>	
Utility Operand	<i>file_name</i>	
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.

ISO/IEC/IEEE 9945:2026(en)

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

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

Volume	1	Base Definitions, Issue 8.....	1
Chapter	1	Introduction.....	3
	1.1	Scope.....	3
	1.2	Word Usage.....	4
	1.3	Conformance.....	4
	1.4	Normative References.....	5
	1.5	Change History.....	5
	1.6	Terminology.....	5
	1.7	Definitions and Concepts.....	6
	1.8	Portability.....	7
	1.8.1	Codes.....	7
	1.8.2	Margin Code Notation.....	12
Chapter	2	Conformance.....	15
	2.1	Implementation Conformance.....	15
	2.1.1	Requirements.....	15
	2.1.2	Documentation.....	16
	2.1.3	POSIX Conformance.....	17
	2.1.4	XSI Conformance.....	19
	2.1.5	Option Groups.....	20
	2.1.6	Options.....	25
	2.2	Application Conformance.....	27
	2.2.1	Strictly Conforming POSIX Application.....	27
	2.2.2	Conforming POSIX Application.....	28
	2.2.3	Conforming POSIX Application Using Extensions.....	28
	2.2.4	Strictly Conforming XSI Application.....	29
	2.2.5	Conforming XSI Application Using Extensions.....	29
	2.3	Language-Dependent Services for the C Programming Language.....	29
	2.4	Other Language-Related Specifications.....	29
Chapter	3	Definitions.....	31
	3.1	Abortive Release.....	31
	3.2	Absolute Pathname.....	31
	3.3	Access Mode.....	31
	3.4	Additional File Access Control Mechanism.....	31
	3.5	Address Space.....	31
	3.6	Advisory Information.....	31
	3.7	Affirmative Response.....	32
	3.8	Alert.....	32
	3.9	Alert Character (<alert>).....	32
	3.10	Alias Name.....	32
	3.11	Alignment.....	32

3.12	Alternate File Access Control Mechanism	32
3.13	Alternate Signal Stack.....	33
3.14	Ancillary Data.....	33
3.15	Angle Brackets.....	33
3.16	Anonymous Memory Object.....	33
3.17	Apostrophe Character (<apostrophe>).....	33
3.18	Application.....	33
3.19	Application Address.....	33
3.20	Application Program Interface (API).....	33
3.21	Appropriate Privileges	34
3.22	Argument	34
3.23	Arm (a Timer)	34
3.24	Asterisk Character (<asterisk>)	34
3.25	Async-Cancel-Safe Function.....	34
3.26	Asynchronous Events.....	34
3.27	Asynchronous Input and Output	34
3.28	Async-Signal-Safe Function.....	35
3.29	Asynchronously-Generated Signal.....	35
3.30	Asynchronous I/O Completion.....	35
3.31	Asynchronous I/O Operation.....	35
3.32	Atomic Operation	35
3.33	Authentication.....	35
3.34	Authorization	36
3.35	Background Job	36
3.36	Background Process.....	36
3.37	Background Process Group	36
3.38	Backquote Character.....	36
3.39	Backslash Character (<backslash>)	36
3.40	Backspace Character (<backspace>)	37
3.41	Barrier	37
3.42	Basename.....	37
3.43	Basic Regular Expression (BRE).....	37
3.44	Bind	37
3.45	Blank Character (<blank>).....	37
3.46	Blank Line.....	37
3.47	Blocked Process (or Thread)	37
3.48	Blocking.....	38
3.49	Block-Mode Terminal	38
3.50	Block Special File.....	38
3.51	Braces	38
3.52	Brackets.....	38
3.53	Broadcast	38
3.54	Built-In Utility (or Built-In).....	39
3.55	Byte.....	39
3.56	Byte Input/Output Functions.....	39
3.57	Carriage-Return Character (<carriage-return>)	39
3.58	Character	39
3.59	Character Array.....	40
3.60	Character Class.....	40
3.61	Character Set.....	40
3.62	Character Special File	40
3.63	Character String.....	40

3.64	Child Process	40
3.65	Circumflex Character (<circumflex>).....	40
3.66	Clock	41
3.67	Clock Jump.....	41
3.68	Clock Tick.....	41
3.69	Code Block	41
3.70	Coded Character Set	41
3.71	Codeset	41
3.72	Collating Element.....	41
3.73	Collation	42
3.74	Collation Sequence.....	42
3.75	Column Position.....	42
3.76	Command.....	42
3.77	Command Language Interpreter	42
3.78	Composite Graphic Symbol.....	43
3.79	Condition Variable	43
3.80	Connected Socket	43
3.81	Connection	43
3.82	Connection Mode.....	43
3.83	Connectionless Mode	43
3.84	Control Character.....	43
3.85	Control Operator	44
3.86	Controlling Process.....	44
3.87	Controlling Terminal	44
3.88	Conversion Descriptor	44
3.89	Core Image	44
3.90	CPU Time (Execution Time)	44
3.91	CPU-Time Clock.....	44
3.92	CPU-Time Timer.....	45
3.93	Current Job	45
3.94	Current Working Directory.....	45
3.95	Cursor Position.....	45
3.96	Datagram	45
3.97	Data Race.....	45
3.98	Data Segment.....	45
3.99	Decimal-Point Character	45
3.100	Declaration Utility.....	45
3.101	Device	46
3.102	Device ID	46
3.103	Directory	46
3.104	Directory Entry (or Hard Link).....	46
3.105	Directory Stream	46
3.106	Disarm (a Timer)	46
3.107	Display.....	46
3.108	Display Line.....	46
3.109	Dollar-Sign Character (<dollar-sign>)	46
3.110	Dot.....	47
3.111	Dot-Dot.....	47
3.112	Dot-Po File.....	47
3.113	Double-Quote Character	47
3.114	Downshifting	47
3.115	Driver	47

3.116	Effective Group ID	47
3.117	Effective User ID	47
3.118	Eight-Bit Transparency	48
3.119	Empty Directory	48
3.120	Empty Line.....	48
3.121	Empty String (or Null String).....	48
3.122	Empty Wide-Character String.....	48
3.123	Encoding Rule	48
3.124	Entire Regular Expression.....	48
3.125	Epoch	48
3.126	Equivalence Class.....	49
3.127	Era.....	49
3.128	Event Management	49
3.129	Executable File.....	49
3.130	Execute.....	49
3.131	Execution Time	49
3.132	Execution Time Monitoring.....	49
3.133	Expand.....	50
3.134	Extended Regular Expression (ERE).....	50
3.135	Extended Security Controls	50
3.136	Feature Test Macro	50
3.137	Field.....	50
3.138	FIFO Special File (or FIFO)	51
3.139	File	51
3.140	File Description	51
3.141	File Descriptor	51
3.142	File Group Class	51
3.143	File Lock	51
3.144	File Mode.....	52
3.145	File Mode Bits	52
3.146	Filename	52
3.147	Filename String.....	52
3.148	File Offset	52
3.149	File Other Class	52
3.150	File Owner Class	52
3.151	File Permission Bits.....	53
3.152	File Serial Number	53
3.153	File System	53
3.154	File Type	53
3.155	Filter	53
3.156	First Open (of a File)	53
3.157	Flow Control	53
3.158	Foreground Job.....	54
3.159	Foreground Process	54
3.160	Foreground Process Group.....	54
3.161	Foreground Process Group ID	54
3.162	Form-Feed Character (<form-feed>).....	54
3.163	Graphic Character	54
3.164	Group Database.....	55
3.165	Group ID.....	55
3.166	Group Name	55
3.167	Hard Limit.....	55

3.168	Hard Link	55
3.169	Hole	55
3.170	Home Directory	56
3.171	Host Byte Order.....	56
3.172	Incomplete Line.....	56
3.173	Inf.....	56
3.174	Interactive Device.....	56
3.175	Interactive Shell	56
3.176	Internationalization	56
3.177	Interprocess Communication	56
3.178	Intrinsic Utility	57
3.179	Invoke	57
3.180	Job	57
3.181	Job Control	57
3.182	Job ID	57
3.183	Joinable Thread.....	58
3.184	Last Close (of a File).....	58
3.185	Line.....	58
3.186	Linger.....	58
3.187	Link	58
3.188	Link Count	58
3.189	Live Process.....	59
3.190	Live Thread	59
3.191	Local Customs	59
3.192	Local Interprocess Communication (Local IPC).....	59
3.193	Locale	59
3.194	Localization.....	59
3.195	Lock-Free Operation	59
3.196	Login	60
3.197	Login Name	60
3.198	Map	60
3.199	Matched	60
3.200	Memory Mapped Files	60
3.201	Memory Object	60
3.202	Memory-Resident.....	60
3.203	Message	61
3.204	Message Catalog.....	61
3.205	Message Catalog Descriptor	61
3.206	Message Queue.....	61
3.207	Messages Object	61
3.208	Mode	61
3.209	Monotonic Clock	61
3.210	Mount Point	62
3.211	Multi-Character Collating Element	62
3.212	Multi-Threaded Library	62
3.213	Multi-Threaded Process	62
3.214	Multi-Threaded Program.....	62
3.215	Mutex	62
3.216	Name.....	63
3.217	NaN (Not a Number)	63
3.218	Native Language	63
3.219	Negative	63

3.220	Negative Response.....	63
3.221	Network.....	63
3.222	Network Address.....	63
3.223	Network Byte Order.....	64
3.224	Newline Character (<newline>).....	64
3.225	Nice Value.....	64
3.226	Non-Blocking.....	64
3.227	Non-Spacing Characters.....	64
3.228	NUL.....	64
3.229	Null Byte.....	65
3.230	Null Pointer.....	65
3.231	Null String.....	65
3.232	Null Terminator.....	65
3.233	Null Wide-Character Code.....	65
3.234	Number-Sign Character (<number-sign>).....	65
3.235	Object File.....	65
3.236	Octet.....	65
3.237	OFD-Owned File Lock.....	66
3.238	Offset Maximum.....	66
3.239	Opaque Address.....	66
3.240	Open File.....	66
3.241	Open File Description.....	66
3.242	Operand.....	66
3.243	Operator.....	66
3.244	Option.....	67
3.245	Option-Argument.....	67
3.246	Orientation.....	67
3.247	Orphaned Process Group.....	67
3.248	Page.....	67
3.249	Page Size.....	67
3.250	Parameter.....	68
3.251	Parent Directory.....	68
3.252	Parent Process.....	68
3.253	Parent Process ID.....	68
3.254	Pathname.....	68
3.255	Pathname Component.....	69
3.256	Path Prefix.....	69
3.257	Pattern.....	69
3.258	Period Character (<period>).....	69
3.259	Permissions.....	69
3.260	Persistence.....	69
3.261	Pipe.....	70
3.262	Polling.....	70
3.263	Portable Character Set.....	70
3.264	Portable Filename.....	70
3.265	Portable Filename Character Set.....	70
3.266	Portable Messages Object Source File (or Dot-Po File).....	70
3.267	Positional Parameter.....	71
3.268	Positive.....	71
3.269	Preallocation.....	71
3.270	Preempted Process (or Thread).....	71

3.271	Previous Job	71
3.272	Printable Character	71
3.273	Printable File	71
3.274	Priority	72
3.275	Priority Inversion	72
3.276	Priority Scheduling	72
3.277	Priority-Based Scheduling	72
3.278	Privilege	72
3.279	Process	72
3.280	Process Group	72
3.281	Process Group ID	72
3.282	Process Group Leader	72
3.283	Process Group Lifetime	73
3.284	Process ID	73
3.285	Process Lifetime	73
3.286	Process Memory Locking	73
3.287	Process Termination	73
3.288	Process Virtual Time	74
3.289	Process-Owned File Lock	74
3.290	Process-To-Process Communication	74
3.291	Program	74
3.292	Protocol	74
3.293	Pseudo-Terminal	74
3.294	Radix Character (or Decimal-Point Character)	74
3.295	Read-Only File System	75
3.296	Read-Write Lock	75
3.297	Real Group ID	75
3.298	Real Time	75
3.299	Realtime Signal Extension	75
3.300	Real User ID	75
3.301	Record	75
3.302	Record Lock	76
3.303	Redirection	76
3.304	Redirection Operator	76
3.305	Referenced Shared Memory Object	76
3.306	Refresh	76
3.307	Regular Built-In Utility (or Regular Built-In)	76
3.308	Regular Expression	76
3.309	Region	76
3.310	Regular File	77
3.311	Relative Pathname	77
3.312	Relocatable File	77
3.313	Relocation	77
3.314	(Time) Resolution	77
3.315	Robust Mutex	77
3.316	Root Directory	77
3.317	Runnable Process (or Thread)	77
3.318	Running Process (or Thread)	77
3.319	Saved Resource Limits	78
3.320	Saved Set-Group-ID	78
3.321	Saved Set-User-ID	78
3.322	Scheduling	78

3.323	Scheduling Allocation Domain	78
3.324	Scheduling Contention Scope	78
3.325	Scheduling Policy	79
3.326	Screen	79
3.327	Scroll	79
3.328	Semaphore	79
3.329	Session	79
3.330	Session Leader	79
3.331	Session Lifetime	79
3.332	Shared Memory Object	80
3.333	Shell	80
3.334	Shell, the	80
3.335	Shell Script	80
3.336	Signal	80
3.337	Signal Stack	80
3.338	Single-Quote Character	80
3.339	Single-Threaded Process	80
3.340	Single-Threaded Program	81
3.341	Slash Character (<slash>)	81
3.342	Socket	81
3.343	Socket Address	81
3.344	Soft Limit	81
3.345	Source Code	81
3.346	Space Character (<space>)	82
3.347	Sparse File	82
3.348	Spawn	82
3.349	Special Built-In Utility (or Special Built-In)	82
3.350	Special Parameter	82
3.351	Spin Lock	82
3.352	Sporadic Server	82
3.353	Standard Error	82
3.354	Standard Input	83
3.355	Standard Output	83
3.356	Standard Utilities	83
3.357	Stream	83
3.358	String	83
3.359	Subshell	84
3.360	Successfully Transferred	84
3.361	Supplementary Group ID	84
3.362	Suspended Job	84
3.363	Symbolic Constant	84
3.364	Symbolic Link	85
3.365	Synchronization Operation	85
3.366	Synchronized Input and Output	85
3.367	Synchronized I/O Completion	85
3.368	Synchronized I/O Data Integrity Completion	85
3.369	Synchronized I/O File Integrity Completion	85
3.370	Synchronized I/O Operation	86
3.371	Synchronous I/O Operation	86
3.372	Synchronously-Generated Signal	86
3.373	System	86
3.374	System Boot	86

3.375	System Clock.....	86
3.376	System Console	86
3.377	System Crash	86
3.378	System Databases.....	87
3.379	System Documentation	87
3.380	System Process.....	87
3.381	System Reboot	87
3.382	System-Wide	87
3.383	Tab Character (<tab>).....	87
3.384	Terminal (or Terminal Device)	87
3.385	Text Column.....	87
3.386	Text Domain.....	88
3.387	Text File.....	88
3.388	Thread.....	88
3.389	Thread ID	88
3.390	Thread Lifetime	89
3.391	Thread List	89
3.392	Thread Termination	89
3.393	Thread-Safe	89
3.394	Thread-Specific Data Key.....	89
3.395	Tilde Character (<tilde>).....	90
3.396	Timeouts	90
3.397	Timer	90
3.398	Timer Overrun.....	90
3.399	Token.....	90
3.400	Typed Memory Name Space	90
3.401	Typed Memory Object.....	90
3.402	Typed Memory Pool	90
3.403	Typed Memory Port.....	91
3.404	Unbind.....	91
3.405	Unit Data	91
3.406	Upshifting	91
3.407	User Database	91
3.408	User ID.....	91
3.409	User Name	91
3.410	Utility	92
3.411	Variable.....	92
3.412	Vertical-Tab Character (<vertical-tab>).....	92
3.413	White Space.....	92
3.414	White-Space Byte	92
3.415	White-Space Character	92
3.416	White-Space Wide Character.....	92
3.417	Wide-Character Code (C Language)	93
3.418	Wide-Character Input/Output Functions	93
3.419	Wide-Character String.....	93
3.420	Word.....	93
3.421	Working Directory (or Current Working Directory)	93
3.422	Worldwide Portability Interface	93
3.423	Write.....	93
3.424	XSI	93
3.425	XSI-Conformant	94
3.426	Zombie Process.....	94

	3.427	Zombie Thread	94
	3.428	±0	94
Chapter	4	General Concepts	95
	4.1	Case Insensitive Comparisons	95
	4.2	Concurrent Execution	95
	4.3	Default Initialization	95
	4.4	Directory Operations	96
	4.5	Directory Protection	96
	4.6	Extended Security Controls	96
	4.7	File Access Permissions	97
	4.8	File Hierarchy	97
	4.9	Filenames	97
	4.10	Filename Portability	98
	4.11	File System Cache	98
	4.12	File Times Update	98
	4.13	Host and Network Byte Orders	99
	4.14	Measurement of Execution Time	99
	4.15	Memory Ordering and Synchronization	100
	4.15.1	Memory Ordering	100
	4.15.2	Memory Synchronization	104
	4.16	Pathname Resolution	105
	4.17	Process ID Reuse	106
	4.18	Scheduling Policy	107
	4.19	Seconds Since the Epoch	107
	4.20	Semaphore	108
	4.21	Special Device Drivers	108
	4.22	Thread-Safety	108
	4.23	Treatment of Error Conditions for Mathematical Functions	109
	4.23.1	Domain Error	109
	4.23.2	Pole Error	109
	4.23.3	Range Error	110
	4.24	Treatment of NaN Arguments for the Mathematical Functions	110
	4.25	Utility	111
	4.26	Variable Assignment	111
Chapter	5	File Format Notation	113
Chapter	6	Character Set	117
	6.1	Portable Character Set	117
	6.2	Character Encoding	120
	6.3	C Language Wide-Character Codes	120
	6.4	Character Set Description File	121
	6.4.1	State-Dependent Character Encodings	125
Chapter	7	Locale	127
	7.1	General	127
	7.2	POSIX Locale	128
	7.3	Locale Definition	128
	7.3.1	LC_CTYPE	131

	7.3.2	LC_COLLATE.....	139
	7.3.3	LC_MONETARY.....	147
	7.3.4	LC_NUMERIC.....	151
	7.3.5	LC_TIME.....	152
	7.3.6	LC_MESSAGES.....	159
	7.4	Locale Definition Grammar.....	160
	7.4.1	Locale Lexical Conventions.....	160
	7.4.2	Locale Grammar.....	161
Chapter	8	Environment Variables.....	167
	8.1	Environment Variable Definition.....	167
	8.2	Internationalization Variables.....	169
	8.3	Other Environment Variables.....	174
Chapter	9	Regular Expressions.....	179
	9.1	Regular Expression Definitions.....	179
	9.2	Regular Expression General Requirements.....	180
	9.3	Basic Regular Expressions.....	181
	9.3.1	BREs Matching a Single Character or Collating Element.....	181
	9.3.2	BRE Ordinary Characters.....	181
	9.3.3	BRE Special Characters.....	182
	9.3.4	Periods in BREs.....	182
	9.3.5	RE Bracket Expression.....	182
	9.3.6	BREs Matching Multiple Characters.....	185
	9.3.7	BRE Precedence.....	186
	9.3.8	BRE Expression Anchoring.....	186
	9.4	Extended Regular Expressions.....	187
	9.4.1	EREs Matching a Single Character or Collating Element.....	187
	9.4.2	ERE Ordinary Characters.....	187
	9.4.3	ERE Special Characters.....	188
	9.4.4	Periods in EREs.....	188
	9.4.5	ERE Bracket Expression.....	188
	9.4.6	EREs Matching Multiple Characters.....	189
	9.4.7	ERE Alternation.....	190
	9.4.8	ERE Precedence.....	190
	9.4.9	ERE Expression Anchoring.....	190
	9.5	Regular Expression Grammar.....	191
	9.5.1	BRE/ERE Grammar Lexical Conventions.....	191
	9.5.2	RE and Bracket Expression Grammar.....	192
	9.5.3	ERE Grammar.....	194
Chapter	10	Directory Structure and Devices.....	197
	10.1	Directory Structure and Files.....	197
	10.2	Output Devices and Terminal Types.....	197
Chapter	11	General Terminal Interface.....	199
	11.1	Interface Characteristics.....	199
	11.1.1	Opening a Terminal Device File.....	199
	11.1.2	Process Groups.....	199
	11.1.3	The Controlling Terminal.....	200

	11.1.4	Terminal Access Control	200
	11.1.5	Input Processing and Reading Data	201
	11.1.6	Canonical Mode Input Processing.....	202
	11.1.7	Non-Canonical Mode Input Processing.....	202
	11.1.8	Writing Data and Output Processing	203
	11.1.9	Special Characters	203
	11.1.10	Modem Disconnect	205
	11.1.11	Closing a Terminal Device File.....	205
	11.2	Parameters that Can be Set	205
	11.2.1	The termios Structure	205
	11.2.2	Input Modes.....	206
	11.2.3	Output Modes.....	207
	11.2.4	Control Modes	209
	11.2.5	Local Modes.....	210
	11.2.6	Special Control Characters.....	212
Chapter	12	Utility Conventions.....	213
	12.1	Utility Argument Syntax.....	213
	12.2	Utility Syntax Guidelines.....	215
Chapter	13	Namespace and Future Directions	219
Chapter	14	Headers	221
Volume	2	System Interfaces, Issue 8.....	491
Chapter	1	Introduction.....	493
	1.1	Relationship to Other Formal Standards.....	493
	1.2	Format of Entries.....	493
Chapter	2	General Information	495
	2.1	Use and Implementation of Interfaces.....	495
	2.1.1	Use and Implementation of Functions.....	495
	2.1.2	Use and Implementation of Macros	496
	2.2	The Compilation Environment	496
	2.2.1	POSIX.1 Symbols.....	496
	2.2.2	The Name Space.....	498
	2.3	Error Numbers.....	507
	2.3.1	Additional Error Numbers	513
	2.4	Signal Concepts	513
	2.4.1	Signal Generation and Delivery.....	513
	2.4.2	Realtime Signal Generation and Delivery	515
	2.4.3	Signal Actions	516
	2.4.4	Signal Effects on Other Functions.....	520
	2.5	Standard I/O Streams	521
	2.5.1	Interaction of File Descriptors and Standard I/O Streams	522
	2.5.2	Stream Orientation and Encoding Rules	524
	2.6	File Descriptor Allocation	525
	2.7	XSI Interprocess Communication	526
	2.7.1	IPC General Description	526

	2.8	Realtime	527
	2.8.1	Realtime Signals	528
	2.8.2	Asynchronous I/O	528
	2.8.3	Memory Management	529
	2.8.4	Process Scheduling.....	531
	2.8.5	Clocks and Timers	535
	2.9	Threads	537
	2.9.1	Thread-Safety.....	537
	2.9.2	Thread IDs.....	538
	2.9.3	Thread Mutexes	539
	2.9.4	Thread Scheduling	540
	2.9.5	Thread Cancellation.....	542
	2.9.6	Thread Read-Write Locks.....	547
	2.9.7	Thread Interactions with File Operations.....	547
	2.9.8	Use of Application-Managed Thread Stacks.....	548
	2.9.9	Synchronization Object Copies and Alternative Mappings.....	548
	2.10	Sockets	549
	2.10.1	Address Families	549
	2.10.2	Addressing	549
	2.10.3	Protocols	549
	2.10.4	Routing	550
	2.10.5	Interfaces	550
	2.10.6	Socket Types.....	550
	2.10.7	Socket I/O Mode.....	551
	2.10.8	Socket Owner.....	551
	2.10.9	Socket Queue Limits.....	551
	2.10.10	Pending Error	551
	2.10.11	Socket Receive Queue.....	552
	2.10.12	Socket Out-of-Band Data State.....	552
	2.10.13	Connection Indication Queue	553
	2.10.14	Signals.....	553
	2.10.15	Asynchronous Errors.....	553
	2.10.16	Use of Options	554
	2.10.17	Use of Sockets for Local UNIX Connections	557
	2.10.18	Use of Sockets over Internet Protocols.....	558
	2.10.19	Use of Sockets over Internet Protocols Based on IPv4.....	558
	2.10.20	Use of Sockets over Internet Protocols Based on IPv6.....	558
	2.11	Data Types.....	561
	2.11.1	Defined Types	562
	2.11.2	The char Type.....	563
	2.12	Status Information	563
Chapter	3	System Interfaces.....	565
Volume	3	Shell and Utilities, Issue 8	2451
Chapter	1	Introduction.....	2453
	1.1	Relationship to Other Documents	2453

1.1.1	System Interfaces.....	2453
1.1.2	Concepts Derived from the ISO C Standard	2457
1.2	Utility Limits.....	2459
1.3	Grammar Conventions.....	2461
1.4	Utility Description Defaults.....	2462
1.5	Considerations for Utilities in Support of Files of Arbitrary Size.....	2469
1.6	Built-In Utilities	2470
1.7	Intrinsic Utilities.....	2470
Chapter 2	Shell Command Language	2472
2.1	Shell Introduction.....	2472
2.2	Quoting.....	2472
2.2.1	Escape Character (Backslash).....	2473
2.2.2	Single-Quotes.....	2473
2.2.3	Double-Quotes.....	2473
2.2.4	Dollar-Single-Quotes	2474
2.3	Token Recognition.....	2475
2.3.1	Alias Substitution.....	2477
2.4	Reserved Words.....	2478
2.5	Parameters and Variables.....	2478
2.5.1	Positional Parameters	2479
2.5.2	Special Parameters	2479
2.5.3	Shell Variables.....	2481
2.6	Word Expansions	2483
2.6.1	Tilde Expansion	2485
2.6.2	Parameter Expansion.....	2485
2.6.3	Command Substitution	2489
2.6.4	Arithmetic Expansion.....	2490
2.6.5	Field Splitting	2491
2.6.6	Pathname Expansion	2493
2.6.7	Quote Removal.....	2493
2.7	Redirection	2493
2.7.1	Redirecting Input	2494
2.7.2	Redirecting Output	2494
2.7.3	Appending Redirected Output	2495
2.7.4	Here-Document	2495
2.7.5	Duplicating an Input File Descriptor	2497
2.7.6	Duplicating an Output File Descriptor	2497
2.7.7	Open File Descriptors for Reading and Writing.....	2497
2.8	Exit Status and Errors	2497
2.8.1	Consequences of Shell Errors	2497
2.8.2	Exit Status for Commands	2499
2.9	Shell Commands	2499
2.9.1	Simple Commands.....	2500
2.9.2	Pipelines	2504
2.9.3	Lists	2505
2.9.4	Compound Commands.....	2508
2.9.5	Function Definition Command	2511
2.10	Shell Grammar.....	2512
2.10.1	Shell Grammar Lexical Conventions.....	2512
2.10.2	Shell Grammar Rules.....	2513

2.11	Job Control	2518
2.12	Signals and Error Handling.....	2521
2.13	Shell Execution Environment.....	2522
2.14	Pattern Matching Notation	2523
2.14.1	Patterns Matching a Single Character.....	2523
2.14.2	Patterns Matching Multiple Characters.....	2524
2.14.3	Patterns Used for Filename Expansion.....	2525
2.15	Special Built-In Utilities.....	2526
Chapter 3	Utilities.....	2573
Volume 4	Rationale (Informative), Issue 8.....	3633
Part A	Base Definitions	3635
Appendix A	Rationale for Base Definitions.....	3637
A.1	Introduction	3637
A.1.1	Scope	3637
A.1.2	Word Usage.....	3639
A.1.3	Conformance.....	3639
A.1.4	Normative References	3639
A.1.5	Change History	3639
A.1.6	Terminology	3639
A.1.7	Definitions and Concepts.....	3642
A.1.8	Portability.....	3642
A.2	Conformance.....	3643
A.2.1	Implementation Conformance	3643
A.2.2	Application Conformance.....	3647
A.2.3	Language-Dependent Services for the C Programming Language.....	3648
A.2.4	Other Language-Related Specifications.....	3648
A.3	Definitions	3648
A.4	General Concepts	3676
A.4.1	Case Insensitive Comparisons	3676
A.4.2	Concurrent Execution.....	3676
A.4.3	Default Initialization.....	3677
A.4.4	Directory Operations	3677
A.4.5	Directory Protection.....	3677
A.4.6	Extended Security Controls.....	3677
A.4.7	File Access Permissions.....	3677
A.4.8	File Hierarchy	3678
A.4.9	Filenames.....	3678
A.4.10	Filename Portability.....	3679
A.4.11	File System Cache	3680
A.4.12	File Times Update	3680
A.4.13	Host and Network Byte Order	3681
A.4.14	Measurement of Execution Time	3681
A.4.15	Memory Ordering and Synchronization	3681
A.4.16	Pathname Resolution.....	3683
A.4.17	Process ID Reuse	3685
A.4.18	Scheduling Policy.....	3685

A.4.19	Seconds Since the Epoch	3685
A.4.20	Semaphore.....	3686
A.4.21	Special Device Drivers.....	3686
A.4.22	Thread-Safety.....	3687
A.4.23	Treatment of Error Conditions for Mathematical Functions	3687
A.4.24	Treatment of NaN Arguments for Mathematical Functions	3687
A.4.25	Utility	3687
A.4.26	Variable Assignment.....	3687
A.5	File Format Notation	3688
A.6	Character Set.....	3688
A.6.1	Portable Character Set	3688
A.6.2	Character Encoding	3689
A.6.3	C Language Wide-Character Codes	3689
A.6.4	Character Set Description File	3690
A.7	Locale	3692
A.7.1	General.....	3692
A.7.2	POSIX Locale	3693
A.7.3	Locale Definition	3693
A.7.4	Locale Definition Grammar	3701
A.7.5	Locale Definition Example.....	3701
A.8	Environment Variables	3704
A.8.1	Environment Variable Definition.....	3704
A.8.2	Internationalization Variables	3705
A.8.3	Other Environment Variables.....	3706
A.9	Regular Expressions	3709
A.9.1	Regular Expression Definitions.....	3709
A.9.2	Regular Expression General Requirements.....	3710
A.9.3	Basic Regular Expressions	3711
A.9.4	Extended Regular Expressions.....	3715
A.9.5	Regular Expression Grammar	3716
A.10	Directory Structure and Devices.....	3717
A.10.1	Directory Structure and Files.....	3717
A.10.2	Output Devices and Terminal Types.....	3717
A.11	General Terminal Interface	3718
A.11.1	Interface Characteristics.....	3719
A.11.2	Parameters that Can be Set	3723
A.12	Utility Conventions.....	3724
A.12.1	Utility Argument Syntax.....	3724
A.12.2	Utility Syntax Guidelines.....	3725
A.13	Namespace and Future Directions	3728
A.14	Headers.....	3728
A.14.1	Format of Entries.....	3728
A.14.2	Removed Headers in Issue 8	3728
Part	B System Interfaces.....	3729
Appendix	B Rationale for System Interfaces.....	3731
B.1	Introduction	3731
B.1.1	Change History	3731

B.1.2	Relationship to Other Formal Standards	3735
B.1.3	Format of Entries	3735
B.2	General Information	3735
B.2.1	Use and Implementation of Interfaces	3735
B.2.2	The Compilation Environment	3737
B.2.3	Error Numbers.....	3742
B.2.4	Signal Concepts	3746
B.2.5	Standard I/O Streams	3757
B.2.6	File Descriptor Allocation	3758
B.2.7	XSI Interprocess Communication	3758
B.2.8	Realtime	3759
B.2.9	Threads	3806
B.2.10	Sockets	3835
B.2.11	Data Types.....	3837
B.2.12	Status Information	3840
B.3	System Interfaces.....	3840
B.3.1	System Interfaces Removed in this Version	3840
B.3.2	System Interfaces Removed in the Previous Version.....	3842
B.3.3	Examples for Spawn	3842
Part C	Shell and Utilities	3853
Appendix C	Rationale for Shell and Utilities.....	3855
C.1	Introduction	3855
C.1.1	Change History	3855
C.1.2	Relationship to Other Documents	3856
C.1.3	Utility Limits.....	3857
C.1.4	Grammar Conventions.....	3860
C.1.5	Utility Description Defaults.....	3860
C.1.6	Considerations for Utilities in Support of Files of Arbitrary Size	3864
C.1.7	Built-In Utilities	3864
C.1.8	Intrinsic Utilities.....	3865
C.2	Shell Command Language	3866
C.2.1	Shell Introduction.....	3866
C.2.2	Quoting.....	3866
C.2.3	Token Recognition.....	3871
C.2.4	Reserved Words.....	3873
C.2.5	Parameters and Variables.....	3874
C.2.6	Word Expansions	3880
C.2.7	Redirection	3890
C.2.8	Exit Status and Errors	3894
C.2.9	Shell Commands	3895
C.2.10	Shell Grammar.....	3905
C.2.11	Job Control	3906
C.2.12	Signals and Error Handling.....	3907
C.2.13	Shell Execution Environment.....	3907
C.2.14	Pattern Matching Notation	3908
C.2.15	Special Built-In Utilities.....	3912
C.3	Utilities.....	3912
C.3.1	Utilities Removed in this Version	3912

	C.3.2	Utilities Removed in the Previous Version.....	3912
	C.3.3	Exclusion of Utilities.....	3913
Part	D	Portability Considerations.....	3917
Appendix	D	Portability Considerations (Informative).....	3919
	D.1	User Requirements.....	3919
	D.1.1	Configuration Interrogation	3920
	D.1.2	Process Management	3920
	D.1.3	Access to Data.....	3920
	D.1.4	Access to the Environment	3920
	D.1.5	Access to Determinism and Performance Enhancements.....	3920
	D.1.6	Operating System-Dependent Profile	3921
	D.1.7	I/O Interaction	3921
	D.1.8	Internationalization Interaction	3921
	D.1.9	C-Language Extensions.....	3921
	D.1.10	Command Language	3921
	D.1.11	Interactive Facilities	3921
	D.1.12	Accomplish Multiple Tasks Simultaneously	3921
	D.1.13	Complex Data Manipulation.....	3922
	D.1.14	File Hierarchy Manipulation	3922
	D.1.15	Locale Configuration	3922
	D.1.16	Inter-User Communication.....	3922
	D.1.17	System Environment	3922
	D.1.18	Printing.....	3922
	D.1.19	Software Development.....	3922
	D.2	Portability Capabilities.....	3923
	D.2.1	Configuration Interrogation	3923
	D.2.2	Process Management	3924
	D.2.3	Access to Data.....	3924
	D.2.4	Access to the Environment	3925
	D.2.5	Bounded (Realtime) Response	3926
	D.2.6	Operating System-Dependent Profile	3926
	D.2.7	I/O Interaction	3926
	D.2.8	Internationalization Interaction	3927
	D.2.9	C-Language Extensions.....	3927
	D.2.10	Command Language	3927
	D.2.11	Interactive Facilities	3928
	D.2.12	Accomplish Multiple Tasks Simultaneously	3928
	D.2.13	Complex Data Manipulation.....	3928
	D.2.14	File Hierarchy Manipulation	3929
	D.2.15	Locale Configuration	3929
	D.2.16	Inter-User Communication.....	3929
	D.2.17	System Environment	3930
	D.2.18	Printing.....	3930
	D.2.19	Software Development.....	3930
	D.2.20	Future Growth.....	3930
	D.3	Profiling Considerations	3931
	D.3.1	Configuration Options	3931
	D.3.2	Configuration Options (Shell and Utilities)	3931

D.3.3	Configurable Limits	3932
D.3.4	Configuration Options (System Interfaces).....	3933
D.3.5	Configurable Limits	3937
D.3.6	Optional Behavior	3940
Part E	Subprofiling Considerations.....	3941
Appendix E	Subprofiling Considerations (Informative)	3943
E.1	Subprofiling Option Groups.....	3943
	Index.....	3951
List of Figures		
3-1	pax Format Archive Example.....	3263
B-1	Example of a System with Typed Memory	3777
List of Tables		
3-1	Job ID Formats	58
5-1	Escape Sequences and Associated Actions	113
6-1	Portable Character Set	117
6-2	Non-Portable Control Characters	122
7-1	Valid Character Class Combinations.....	135
10-1	Control Character Names	198
2-1	Value of Level for Socket Options.....	554
2-2	Socket-Level Options.....	555
1-1	Actions when Creating a File that Already Exists.....	2455
1-2	Selected ISO C Standard Operators and Control Flow Keywords	2458
1-3	Utility Limit Minimum Values	2459
1-4	Symbolic Utility Limits	2460
1-5	Intrinsic Utilities	2470
3-1	Expressions in Decreasing Precedence in <i>awk</i>	2608
3-2	Escape Sequences in <i>awk</i>	2616
3-3	Operators in <i>bc</i>	2656
3-4	Programming Environments: Type Sizes	2675
3-5	Programming Environments: <i>c17</i> Arguments	2676
3-6	Threaded Programming Environment: <i>c17</i> Arguments.....	2677
3-7	Compression algorithms, <i>-m</i> option-argument values, and suffixes	2737
3-8	ASCII to EBCDIC Conversion.....	2780
3-9	ASCII to IBM EBCDIC Conversion	2781
3-10	File Utility Output Strings	2933
3-11	Table Size Declarations in <i>lex</i>	3040
3-12	Escape Sequences in <i>lex</i>	3042
3-13	ERE Precedence in <i>lex</i>	3042
3-14	Named Characters in <i>od</i>	3229
3-15	ustar Header Block.....	3268
3-16	ustar <i>mode</i> Field	3269
3-17	Octet-Oriented <i>cpio</i> Archive Entry.....	3272

Contents

3-18	Values for <code>cpio c_mode</code> Field	3273
3-19	Variable Names and Default Headers in <code>ps</code>	3314
3-20	Control Character Names in <code>stty</code>	3409
3-21	Circumflex Control Characters in <code>stty</code>	3410
3-22	<code>uuencode</code> Base64 Values	3512
3-23	Internal Limits in <code>yacc</code>	3626
A-1	Historical Practice for Symbolic Links.....	3672



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.

Referenced Documents

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:

Referenced Documents

- 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].

Referenced Documents

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).



1

Vol. 1:

2

Base Definitions, Issue 8

3

The Open Group

4

The Institute of Electrical and Electronics Engineers, Inc.

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)

- 40 • ISO/IEC 9899: 2018, Programming Languages — C (C17)
 - 41 • ISO/IEC TR 24731-2:2010, Programming languages, their environments and system
 - 42 software interfaces — Extensions to the C library — Part 2: Dynamic Allocation Functions
 - 43 • The Open Group Standard, 2021, Additional APIs for the Base Specifications Issue 8, Part 1
 - 44 • The Open Group Standard, 2022, Additional APIs for the Base Specifications Issue 8, Part 2
- 45 Emphasis has been placed on standardizing existing practice for existing users, with changes
- 46 and additions limited to correcting deficiencies in the following areas:
- 47 • Issues raised by Austin Group defect reports and IEEE Interpretations against IEEE Std
 - 48 1003.1.
 - 49 • Issues raised in corrigenda for The Open Group Standards and working group resolutions
 - 50 from The Open Group
 - 51 • Changes to make the text self-consistent with the additional material merged
 - 52 • Features, marked obsolescent in the base documents, have been considered for removal in
 - 53 this version
 - 54 • Alignment with the ISO/IEC 9899: 2018 standard

55 1.2 Word Usage

56 The word *shall* indicates mandatory requirements strictly to be followed in order to conform to

57 the standard and from which no deviation is permitted (*shall* equals *is required to*).^{1, 2}

58 The word *should* indicates that among several possibilities one is recommended as particularly

59 suitable, without mentioning or excluding others; or that a certain course of action is preferred

60 but not necessarily required (*should* equals *is recommended that*).

61 The word *may* is used to indicate a course of action permissible within the limits of the standard

62 (*may* equals *is permitted to*).

63 The word *can* is used for statements of possibility and capability, whether material, physical, or

64 causal (*can* equals *is able to*).

65 1.3 Conformance

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

67 1. The use of the word *must* is deprecated and cannot be used when stating mandatory requirements; *must* is used only to describe

68 unavoidable situations.

69 2. The use of *will* is deprecated and cannot be used when stating mandatory requirements; *will* is only used in statements of fact.

70 1.4 Normative References

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

77 ISO/IEC 646:1991

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

80 ISO 4217:2015

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

82 ISO 8601-1:2019

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

85 ISO C (C17)

86 ISO/IEC 9899:2018, Programming Languages — C.

87 ISO/IEC 10646:2020

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

89 1.5 Change History

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

92 1.6 Terminology

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

94 **can**

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

98 **implementation-defined**

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

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

106 **legacy**

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

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

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

112 **may**

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

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

118 **shall**

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

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

122 **should**

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

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

129 **undefined**

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

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

136 **unspecified**

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

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

143 **1.7 Definitions and Concepts**

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

145 1.8 Portability

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

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

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

158 1.8.1 Codes

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

160 ADV **Advisory Information**

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

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

166 CD **C-Language Development Utilities**

167 The functionality described is optional.

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

171 CPT **Process CPU-Time Clocks**

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

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

177 CX **Extension to the ISO C standard**

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

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

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

189	DC	Device Control
190		The functionality described is optional. The functionality described is also an extension to the
191		ISO C standard.
192		Where applicable, functions are marked with the DC margin legend in the SYNOPSIS section.
193		Where additional semantics apply to a function, the material is identified by use of the DC
194		margin legend.
195	FR	FORTRAN Runtime Utilities
196		The functionality described is optional.
197		Where applicable, utilities are marked with the FR margin legend in the SYNOPSIS section.
198		Where additional semantics apply to a utility, the material is identified by use of the FR margin
199		legend.
200	FSC	File Synchronization
201		The functionality described is optional. The functionality described is also an extension to the
202		ISO C standard.
203		Where applicable, functions are marked with the FSC margin legend in the SYNOPSIS section.
204		Where additional semantics apply to a function, the material is identified by use of the FSC
205		margin legend.
206	IP6	IPV6
207		The functionality described is optional. The functionality described is also an extension to the
208		ISO C standard.
209		Where applicable, functions are marked with the IP6 margin legend in the SYNOPSIS section.
210		Where additional semantics apply to a function, the material is identified by use of the IP6
211		margin legend.
212	MC1	Non-Robust Mutex Priority Protection or Non-Robust Mutex Priority Inheritance or Robust
213		Mutex Priority Protection or Robust Mutex Priority Inheritance
214		The functionality described is optional. The functionality described is also an extension to the
215		ISO C standard.
216		This is a shorthand notation for combinations of multiple option codes.
217		Where applicable, functions are marked with the MC1 margin legend in the SYNOPSIS section.
218		Where additional semantics apply to a function, the material is identified by use of the MC1
219		margin legend.
220		Refer to Section 1.8.2 (on page 12).
221	ML	Process Memory Locking
222		The functionality described is optional. The functionality described is also an extension to the
223		ISO C standard.
224		Where applicable, functions are marked with the ML margin legend in the SYNOPSIS section.
225		Where additional semantics apply to a function, the material is identified by use of the ML
226		margin legend.
227	MLR	Range Memory Locking
228		The functionality described is optional. The functionality described is also an extension to the
229		ISO C standard.
230		Where applicable, functions are marked with the MLR margin legend in the SYNOPSIS section.
231		Where additional semantics apply to a function, the material is identified by use of the MLR
232		margin legend.

233	MSG	Message Passing
234		The functionality described is optional. The functionality described is also an extension to the
235		ISO C standard.
236		Where applicable, functions are marked with the MSG margin legend in the SYNOPSIS section.
237		Where additional semantics apply to a function, the material is identified by use of the MSG
238		margin legend.
239	MX	IEC 60559 Floating-Point
240		The functionality described is optional. The functionality described is mandated by the ISO C
241		standard only for implementations that define <code>__STDC_IEC_559__</code> .
242	MXC	IEC 60559 Complex Floating-Point
243		The functionality described is optional. The functionality described is mandated by the ISO C
244		standard only for implementations that define <code>__STDC_IEC_559_COMPLEX__</code> .
245	MXX	IEC 60559 Floating-Point Extension
246		The functionality described is optional. The functionality described is part of the IEC 60559
247		Floating-Point option, but is an extension to the ISO C standard.
248	OB	Obsolescent
249		The functionality described may be removed in a future version of this volume of POSIX.1-2024.
250		Strictly Conforming POSIX Applications and Strictly Conforming XSI Applications shall not use
251		obsolescent features.
252		Where applicable, the material is identified by use of the OB margin legend.
253	OF	Output Format Incompletely Specified
254		The functionality described is an XSI extension. The format of the output produced by the
255		utility is not fully specified. It is therefore not possible to post-process this output in a consistent
256		fashion. Typical problems include unknown length of strings and unspecified field delimiters.
257		Where applicable, the material is identified by use of the OF margin legend.
258	OH	Optional Header
259		In the SYNOPSIS section of some interfaces in the System Interfaces volume of POSIX.1-2024 an
260		included header is marked as in the following example:
261	OH	<code>#include <sys/types.h></code>
262		<code>#include <fcntl.h></code>
263		<code>int open(const char *path, int oflag, ...);</code>
264		The OH margin legend indicates that the optional header defines constants that will be needed if
265		the function is called with certain flag arguments; thus it may be required for some of the
266		functionality described, but is not needed otherwise.
267	PIO	Prioritized Input and Output
268		The functionality described is optional. The functionality described is also an extension to the
269		ISO C standard.
270		Where applicable, functions are marked with the PIO margin legend in the SYNOPSIS section.
271		Where additional semantics apply to a function, the material is identified by use of the PIO
272		margin legend.
273	PS	Process Scheduling
274		The functionality described is optional. The functionality described is also an extension to the
275		ISO C standard.
276		Where applicable, functions are marked with the PS margin legend in the SYNOPSIS section.
277		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.

323		Where applicable, functions are marked with the SS margin legend in the SYNOPSIS section.
324		Where additional semantics apply to a function, the material is identified by use of the SS
325		margin legend.
326	TCT	Thread CPU-Time Clocks
327		The functionality described is optional. The functionality described is also an extension to the
328		ISO C standard.
329		Where applicable, functions are marked with the TCT margin legend in the SYNOPSIS section.
330		Where additional semantics apply to a function, the material is identified by use of the TCT
331		margin legend.
332	TPI	Non-Robust Mutex Priority Inheritance
333		The functionality described is optional. The functionality described is also an extension to the
334		ISO C standard.
335		Where applicable, functions are marked with the TPI margin legend in the SYNOPSIS section.
336		Where additional semantics apply to a function, the material is identified by use of the TPI
337		margin legend.
338	TPP	Non-Robust Mutex Priority Protection
339		The functionality described is optional. The functionality described is also an extension to the
340		ISO C standard.
341		Where applicable, functions are marked with the TPP margin legend in the SYNOPSIS section.
342		Where additional semantics apply to a function, the material is identified by use of the TPP
343		margin legend.
344	TPS	Thread Execution Scheduling
345		The functionality described is optional. The functionality described is also an extension to the
346		ISO C standard.
347		Where applicable, functions are marked with the TPS margin legend for the SYNOPSIS section.
348		Where additional semantics apply to a function, the material is identified by use of the TPS
349		margin legend.
350	TSA	Thread Stack Address Attribute
351		The functionality described is optional. The functionality described is also an extension to the
352		ISO C standard.
353		Where applicable, functions are marked with the TSA margin legend for the SYNOPSIS section.
354		Where additional semantics apply to a function, the material is identified by use of the TSA
355		margin legend.
356	TSH	Thread Process-Shared Synchronization
357		The functionality described is optional. The functionality described is also an extension to the
358		ISO C standard.
359		Where applicable, functions are marked with the TSH margin legend in the SYNOPSIS section.
360		Where additional semantics apply to a function, the material is identified by use of the TSH
361		margin legend.
362	TSP	Thread Sporadic Server
363		The functionality described is optional. The functionality described is also an extension to the
364		ISO C standard.
365		Where applicable, functions are marked with the TSP margin legend in the SYNOPSIS section.
366		Where additional semantics apply to a function, the material is identified by use of the TSP
367		margin legend.

368	TSS	Thread Stack Size Attribute
369		The functionality described is optional. The functionality described is also an extension to the
370		ISO C standard.
371		Where applicable, functions are marked with the TSS margin legend in the SYNOPSIS section.
372		Where additional semantics apply to a function, the material is identified by use of the TSS
373		margin legend.
374	TYM	Typed Memory Objects
375		The functionality described is optional. The functionality described is also an extension to the
376		ISO C standard.
377		Where applicable, functions are marked with the TYM margin legend in the SYNOPSIS section.
378		Where additional semantics apply to a function, the material is identified by use of the TYM
379		margin legend.
380	UP	User Portability Utilities
381		The functionality described is optional.
382		Where applicable, utilities are marked with the UP margin legend in the SYNOPSIS section.
383		Where additional semantics apply to a utility, the material is identified by use of the UP margin
384		legend.
385	UU	UUCP Utilities
386		The functionality described is optional. The functionality described is also an extension to the
387		ISO C standard.
388		Where applicable, functions are marked with the UU margin legend in the SYNOPSIS section.
389		Where additional semantics apply to a function, the material is identified by use of the UU
390		margin legend.
391	XSI	X/Open System Interfaces
392		The functionality described is part of the X/Open Systems Interfaces option. Functionality
393		marked XSI is an extension to the ISO C standard. Application developers may confidently
394		make use of such extensions on all systems supporting the X/Open System Interfaces option.
395		If an entire SYNOPSIS section is shaded and marked XSI, all the functionality described in that
396		reference page is an extension. See Section 2.1.4 (on page 19).

397 1.8.2 Margin Code Notation

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

401 A Feature Dependent on One or Two Options

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

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

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

407 **A Feature Dependent on Either of the Options Denoted**

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

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

412 **A Feature Dependent on More than Two Options**

413 The following shorthand notations are used:

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

418 **Large Sections Dependent on an Option**

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

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

424

Chapter 2

425

Conformance

426

2.1 Implementation Conformance

427

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

428

429

2.1.1 Requirements

430

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

431

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.

432

433

434

2. The system may support the X/Open System Interfaces (XSI) option as described in [Section 2.1.4](#) (on page 19).

435

436

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.

437

438

439

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:

440

441

- Additional functions

442

- Additional headers

443

- Additional symbols in standard headers

444

- Additional utilities

445

- Additional options for standard utilities

446

- Additional environment variables

447

- Additional file types

448

- Non-conforming file systems (for example, legacy file systems for which `_POSIX_NO_TRUNC` is false, case-insensitive file systems, or network file systems)

449

450

- Dynamically populated file systems (for example, `/proc`)

451

- Additional character special files with special properties (for example, `/dev/stdin`, `/dev/stdout`, and `/dev/stderr`)

452

453

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

454

455

456

457

458

459 Section 2.2.1, on page 27).

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

464 2.1.2 Documentation

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

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

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

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

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

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

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

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

497 **2.1.3 POSIX Conformance**

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

499 *2.1.3.1 POSIX System Interfaces*

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

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

507 The following symbolic constants shall be defined with a value other than `-1`. If a constant

508 is defined with the value zero, applications should use the `sysconf()`, `pathconf()`, or

509 `fpathconf()` functions, or the `getconf` utility, to determine which features are present on the

510 system at that time or for the particular pathname in question.

511 — `_POSIX_CHOWN_RESTRICTED`

512 The use of `chown()` is restricted to a process with appropriate privileges, and to

513 changing the group ID of a file only to the effective group ID of the process or to one

514 of its supplementary group IDs.

515 — `_POSIX_NO_TRUNC`516 Pathname components longer than `{NAME_MAX}` generate an error.

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

518 — Symbolic constants defined with the value 202405L:

519 `_POSIX_ASYNCHRONOUS_IO`

520 `_POSIX_BARRIERS`

521 `_POSIX_CLOCK_SELECTION`

522 `_POSIX_MAPPED_FILES`

523 `_POSIX_MEMORY_PROTECTION`

524 `_POSIX_MONOTONIC_CLOCK`

525 `_POSIX_READER_WRITER_LOCKS`

526 `_POSIX_REALTIME_SIGNALS`

527 `_POSIX_SEMAPHORES`

528 `_POSIX_SPIN_LOCKS`

529 `_POSIX_THREAD_SAFE_FUNCTIONS`

530 `_POSIX_THREADS`

531 `_POSIX_TIMEOUTS`

532 `_POSIX_TIMERS`

533 `_POSIX2_C_BIND`

534 — Symbolic constants defined with a value greater than zero:

535 `_POSIX_JOB_CONTROL`

536 `_POSIX_REGEX`

537 `_POSIX_SAVED_IDS`

538 `_POSIX_SHELL`

539 — Symbolic constants defined with a value other than -1 .

540 `_POSIX_VDISABLE`

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

543 • The system may support one or more options (see [Section 2.1.6](#), on page 25) denoted by the
544 following symbolic constants:

545 `_POSIX_ADVISORY_INFO`
546 `_POSIX_CPUTIME`
547 `_POSIX_DEVICE_CONTROL`
548 `_POSIX_FSYNC`
549 `_POSIX_IPV6`
550 `_POSIX_MEMLOCK`
551 `_POSIX_MEMLOCK_RANGE`
552 `_POSIX_MESSAGE_PASSING`
553 `_POSIX_PRIORITIZED_IO`
554 `_POSIX_PRIORITY_SCHEDULING`
555 `_POSIX_RAW_SOCKETS`
556 `_POSIX_SHARED_MEMORY_OBJECTS`
557 `_POSIX_SPAWN`
558 `_POSIX_SPORADIC_SERVER`
559 `_POSIX_SYNCHRONIZED_IO`
560 `_POSIX_THREAD_ATTR_STACKADDR`
561 `_POSIX_THREAD_CPUTIME`
562 `_POSIX_THREAD_ATTR_STACKSIZE`
563 `_POSIX_THREAD_PRIO_INHERIT`
564 `_POSIX_THREAD_PRIO_PROTECT`
565 `_POSIX_THREAD_PRIORITY_SCHEDULING`
566 `_POSIX_THREAD_PROCESS_SHARED`
567 `_POSIX_THREAD_SPORADIC_SERVER`
568 `_POSIX_TYPED_MEMORY_OBJECTS`
569 `_XOPEN_CRYPT`
570 `_XOPEN_REALTIME`
571 `_XOPEN_REALTIME_THREADS`
572 `_XOPEN_UNIX`

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

575 2.1.3.2 *POSIX Shell and Utilities*

576 The following requirements apply to the shell and utilities:

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

583 POSIX2_C_DEV
 584 POSIX2_CHAR_TERM
 585 POSIX2_FORT_RUN
 586 POSIX2_LOCALEDEF
 587 POSIX2_SW_DEV
 588 POSIX2_UPE
 589 XOPEN_UNIX
 590 XOPEN_UUCP

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

596 2.1.4 XSI Conformance

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

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

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

606 2.1.4.1 XSI System Interfaces

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

- 608 • The system shall support all the functions and headers defined in POSIX.1-2024 as part of
 609 the XSI option denoted by the XSI marking in the SYNOPSIS section, and any extensions
 610 marked with the XSI option marking (see [Section 1.8.1](#), on page 7) within the text.
- 611 • The system shall support the following options defined within POSIX.1-2024 (see [Section](#)
 612 [2.1.6](#), on page 25):

613 `_POSIX_FSYNC`
 614 `_POSIX_THREAD_ATTR_STACKADDR`
 615 `_POSIX_THREAD_ATTR_STACKSIZE`
 616 `_POSIX_THREAD_PROCESS_SHARED`

- 617 • The system may support the following XSI Option Groups (see [Section 2.1.5.2](#), on page 22)
 618 defined within POSIX.1-2024:

619 — Encryption
 620 — Realtime
 621 — Advanced Realtime

- 622 — Realtime Threads
 623 — Advanced Realtime Threads

624 2.1.4.2 XSI Shell and Utilities Conformance

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

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

636	<i>admin</i>	<i>delta</i>	<i>rmdel</i>	<i>val</i>
637	<i>cflow</i>	<i>get</i>	<i>sact</i>	<i>what</i>
638	<i>ctags</i>	<i>nm</i>	<i>sccs</i>	
639	<i>cxref</i>	<i>prs</i>	<i>unget</i>	

640 2.1.5 Option Groups

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

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

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

647 2.1.5.1 Subprofiling Considerations

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

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

656 A Subprofiling Option Group may provide basic system functionality that other Subprofiling
 657 Option Groups and other options depend upon.⁴ If a profile of POSIX.1-2024 does not require an
 658 implementation to provide a Subprofiling Option Group that provides features utilized by a
 659 required Subprofiling Option Group (or option),⁵ the profile shall specify⁶ all of the following:

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

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

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

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

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

- 679 — If the limit is specified as having a fixed value, it shall not be changed by a profile.
- 680 — If a limit is specified as having a minimum or maximum acceptable value, it may be
681 changed by a profile as follows:
 - 682 — A profile may increase a minimum acceptable value, but shall not make a
683 minimum acceptable value smaller.
 - 684 — A profile may reduce a maximum acceptable value, but shall not make a
685 maximum acceptable value larger.
- 686 • A profile shall not change a limit specified as having a minimum or maximum value into a
687 limit specified as having a fixed value.
- 688 • A profile shall not create new limits.
- 689 • Any implementation that conforms to POSIX.1-2024 (including all options and extended
690 limits required by the profile) shall also conform to that profile, except for the possible
691 omission from **<unistd.h>** of a profile support indicator macro required by the profile.

-
- 692 4. As an example, the File System profiling option group provides underlying support for pathname resolution and file creation which are
693 needed by any interface in POSIX.1-2024 that parses a *path* argument. If a profile requires support for the Device Input and Output
694 profiling option group but does not require support for the File System profiling option group, the profile needs to specify how pathname
695 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*
696 argument of `fopen()` when a pathname argument names a file that does not exist), and specify lots of other details.
 - 697 5. As an example, POSIX.1-2024 requires that implementations claiming to support the Range Memory Locking option also support the
698 Process Memory Locking option. A profile could require that the Range Memory Locking option had to be supplied without requiring that
699 the Process Memory Locking option be supplied as long as the profile specifies everything an application developer or system implementor
700 would have to know to build an application or implementation conforming to the profile.
 - 701 6. Note that the profile could just specify that any use of the features not specified by the profile would produce undefined or unspecified
702 results.

703 2.1.5.2 XSI Option Groups

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

708 The following Option Groups are defined.

709 **Encryption**

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

712 OB `crypt()`, `encrypt()`, `setkey()`

713 These functions are marked CRYPT.

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

718 An implementation that claims conformance to this Option Group shall set `_XOPEN_CRYPT` to
 719 a value other than `-1`.

720 **Realtime**

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

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

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

728 An implementation that claims conformance to this Option Group shall set
 729 `_XOPEN_REALTIME` to a value other than `-1`.

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

732 `_POSIX_FSYNC`
 733 `_POSIX_MEMLOCK`
 734 `_POSIX_MEMLOCK_RANGE`
 735 `_POSIX_MESSAGE_PASSING`
 736 `_POSIX_PRIORITIZED_IO`
 737 `_POSIX_PRIORITY_SCHEDULING`
 738 `_POSIX_SHARED_MEMORY_OBJECTS`
 739 `_POSIX_SYNCHRONIZED_IO`

740 If the symbolic constant `_XOPEN_REALTIME` is defined to have a value other than `-1`, then the
 741 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.

781

Realtime Threads

782

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

783

784

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

785

786

```
_POSIX_THREAD_PRIO_INHERIT
```

787

```
_POSIX_THREAD_PRIO_PROTECT
```

788

```
_POSIX_THREAD_PRIORITY_SCHEDULING
```

789

```
_POSIX_THREAD_ROBUST_PRIO_INHERIT
```

790

```
_POSIX_THREAD_ROBUST_PRIO_PROTECT
```

791

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).

792

793

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

794

795

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`:

796

797

```
_POSIX_THREAD_PRIO_INHERIT
```

798

```
_POSIX_THREAD_PRIO_PROTECT
```

799

```
_POSIX_THREAD_PRIORITY_SCHEDULING
```

800

```
_POSIX_THREAD_ROBUST_PRIO_INHERIT
```

801

```
_POSIX_THREAD_ROBUST_PRIO_PROTECT
```

802

Advanced Realtime Threads

803

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

804

805

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.

806

807

808

809

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

810

811

```
_POSIX_THREAD_CPUTIME
```

812

```
_POSIX_THREAD_SPORADIC_SERVER
```

813

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`.

814

815

816

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`:

817

818

```
_POSIX_THREAD_CPUTIME
```

819

```
_POSIX_THREAD_SPORADIC_SERVER
```

820 **2.1.6 Options**

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

825 1. Option not supported for compilation.

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

835 2. Option always supported.

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

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

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

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

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

859 **2.1.6.1 System Interfaces**

860 Refer to `<unistd.h>`, [Constants for Options and Option Groups](#) (on page 458) for the list of
861 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.

905 *ar*
 906 *make*
 907 *nm*
 908 *strip*

909 UP **POSIX2_UPE**

910 The system supports the User Portability Utilities option.

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

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

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

919 XSI **XOPEN_UNIX**

920 The system supports the X/Open System Interfaces (XSI) option (see [Section 2.1.4](#), on page
 921 19).

922 UU **XOPEN_UUCP**

923 The system supports the UUCP Utilities option.

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

925 *uucp*
 926 *uustat*
 927 *uux*

928 **2.2 Application Conformance**

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

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

935 **2.2.1 Strictly Conforming POSIX Application**

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

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

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

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

958 2.2.2 Conforming POSIX Application

959 2.2.2.1 ISO/IEC Conforming POSIX Application

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

964 2.2.2.2 <National Body> Conforming POSIX Application

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

970 2.2.3 Conforming POSIX Application Using Extensions

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

978 2.2.4 Strictly Conforming XSI Application

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

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

999 Within POSIX.1-2024, any restrictions placed upon a Conforming POSIX Application shall
1000 restrict a Strictly Conforming XSI Application.

1001 2.2.5 Conforming XSI Application Using Extensions

1002 A Conforming XSI Application Using Extensions is an application that differs from a Strictly
1003 Conforming XSI Application only in that it uses non-standard facilities that are consistent with
1004 POSIX.1-2024. Such an application shall fully document its requirements for these extended
1005 facilities, in addition to the documentation required of a Strictly Conforming XSI Application.

1006 2.3 Language-Dependent Services for the C Programming Language

1007 Implementors seeking to claim conformance using the ISO C standard shall claim POSIX
1008 conformance as described in [Section 2.1.3](#) (on page 17).

1009 2.4 Other Language-Related Specifications

1010 POSIX.1-2024 is currently specified in terms of the shell command language and ISO C. Bindings
1011 to other programming languages are being developed.

1012 If conformance to POSIX.1-2024 is claimed for implementation of any programming language,
1013 the implementation of that language shall support the use of external symbols distinct to at least
1014 31 bytes in length in the source program text. (That is, identifiers that differ at or before the
1015 thirty-first byte shall be distinct.) If a national or international standard governing a language
1016 defines a maximum length that is less than this value, the language-defined maximum shall be

1017 supported. External symbols that differ only by case shall be distinct when the character set in
1018 use distinguishes uppercase and lowercase characters and the language permits (or requires)
1019 uppercase and lowercase characters to be distinct in external symbols.