

01-december-1997

Information technology - Computer graphics - Graphical Kernel System (GKS) language bindings - Part 4: C (ISO/IEC 8651-4:1995)

Information technology - Computer graphics - Graphical Kernel System (GKS) language bindings - Part 4: C (ISO/IEC 8651-4:1995)

Informationstechnik - Verarbeitung graphischer Daten - Sprachbindungen für das graphische Kernsystem (GKS) - Teil 4: C (ISO/IEC 8651 4:1995)

Technologies de l'information - Infographie - Interfaces langage avec GKS - Partie 4: C
(ISO/IEC 8651-4:1995)

SIST EN ISO/IEC 8651-4:1997

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94->

e283b4ced76d/sist-en-iso-iec-8651-4-1997

Ta slovenski standard je istoveten z: EN ISO/IEC 8651-4:1996

ICS:

| | | |
|--------|--|--|
| 35.060 | Jeziki, ki se uporabljajo v informacijski tehniki in tehnologiji | Languages used in information technology |
| 35.140 | Üæ ~ } æ) ã\ æ! æã æ | Computer graphics |

SIST EN ISO/IEC 8651-4:1997

en

iTeh STANDARD PREVIEW
(standards.iteh.ai)

[SIST EN ISO/IEC 8651-4:1997](https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997)

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997>

EUROPEAN STANDARD

EN ISO/IEC 8651-4

NORME EUROPÉENNE

EUROPÄISCHE NORM

May 1996

ICS 35.060; 35.140

Descriptors: See ISO document

English version

**Information technology - Computer graphics -
Graphical Kernel System (GKS) language bindings
- Part 4: C (ISO/IEC 8651-4:1995)**

Technologies de l'information - Infographie -
Interfaces langage avec GKS - Partie 4: C
(ISO/IEC 8651-4:1995)

Informationstechnik - Verarbeitung graphischer
Daten - Sprachbindungen für das graphische
Kernsystem (GKS) - Teil 4: C (ISO/IEC
8651-4:1995)

(standard.teh.ai)



REPUBLIKA SLOVENIJA
MINISTRSTVO ZA ZNANOST IN TEHNOLOGIJO
Urad RS za standardizacijo in meroslovje
LJUBLJANA

SIST... **EN ISO/IEC 8651-4** ...

PREVZET PO METODI RAZGLASITVE

-2- 1997

This European Standard was approved by CEN on 1996-04-18. CEN members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CEN member.

The European Standards exist in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CEN member into its own language and notified to the Central Secretariat has the same status as the official versions.

CEN members are the national standards bodies of Austria, Belgium, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and United Kingdom.

CEN

European Committee for Standardization
Comité Européen de Normalisation
Europäisches Komitee für Normung

Central Secretariat: rue de Stassart, 36 B-1050 Brussels

Foreword

The text of the International Standard from ISO/IEC/JTC 1 "Information Technology" of the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) has been taken over as a European Standard by the Technical Board of CEN.

This European Standard shall be given the status of a national standard, either by publication of an identical text or by endorsement, at the latest by November 1996, and conflicting national standards shall be withdrawn at the latest by November 1996.

According to the CEN/CENELEC Internal Regulations, the national standards organizations of the following countries are bound to announce this European Standard: Austria, Belgium, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland and the United Kingdom.

Endorsement notice

The text of the International Standard ISO/IEC 8651-4:1995 has been approved by CEN as a European Standard without any modification.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

SIST EN ISO/IEC 8651-4:1997

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997>



INTERNATIONAL STANDARD

ISO/IEC
8651-4

Second edition
1995-06-01

Information technology — Computer graphics — Graphical Kernel System (GKS) language bindings —

Part 4:
(C)

SIST EN ISO/IEC 8651-4:1997

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e28347ed76d7/sist-en-iso-iec-8651-4-1997>

*Technologies de l'information — Infographie — Interfaces langage avec
GKS —*

Partie 4: C



Reference number
ISO/IEC 8651-4:1995(E)

ISO/IEC 8651-4 : 1995(E)

Contents

| | |
|---|----|
| Foreword..... | v |
| Introduction..... | vi |
| 1 Scope..... | 1 |
| 2 Normative references..... | 2 |
| 3 The C language binding..... | 3 |
| 3.1 Classification and designation..... | 3 |
| 3.2 Functions versus macros..... | 3 |
| 3.3 Character strings..... | 3 |
| 3.4 Function identifiers..... | 3 |
| 3.5 Registration..... | 3 |
| 3.6 Identifiers for graphical items..... | 4 |
| 3.7 Return values..... | 4 |
| 3.8 Headers..... | 4 |
| 3.8.1 gks.h | 4 |
| 3.8.2 gks_compat.h | 4 |
| 3.9 Memory management..... | 5 |
| 3.9.1 Functions which return simple lists..... | 5 |
| 3.9.2 Functions which return complex data structures..... | 5 |
| 3.10 Error handling..... | 7 |
| 3.10.1 Application supplied error handlers..... | 7 |
| 3.10.2 Error codes..... | 7 |
| 3.10.3 C-specific GKS errors..... | 7 |
| 3.11 Colour representations and specifications..... | 7 |
| 3.12 Colour characteristics..... | 7 |
| 3.13 Storage of multi-dimensional arrays..... | 8 |
| 3.13.1 Storage of 2*3 matrices..... | 8 |
| 3.13.2 Storage of conics in 3*3 matrices..... | 8 |
| 3.13.3 Storage of colour arrays..... | 8 |
| 3.14 Compatibility with the 1991 edition..... | 8 |
| 4 Tables..... | 9 |
| 4.1 Abbreviation policy in construction of identifiers..... | 9 |
| 4.2 Table of abbreviations used..... | 9 |
| 4.3 Function names..... | 13 |
| 4.3.1 List ordered alphabetically by bound name..... | 13 |
| 4.3.2 List ordered alphabetically by GKS name..... | 20 |
| 5 Type definitions..... | 28 |
| 5.1 Mapping of GKS data types..... | 28 |
| 5.2 Environment-defined type definitions..... | 28 |
| 5.3 Implementation dependent type definitions..... | 29 |
| 5.4 Implementation independent type definitions..... | 35 |
| 6 Macro definitions..... | 91 |
| 6.1 Function identifiers..... | 91 |
| 6.1.1 In order of appearance..... | 91 |
| 6.1.2 In alphabetical order..... | 95 |

© ISO/IEC 1995

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

| | | |
|--------|---|-----|
| 6.2 | Error codes..... | 99 |
| 6.3 | Miscellaneous..... | 104 |
| 6.3.1 | Linetypes..... | 104 |
| 6.3.2 | Marker types..... | 104 |
| 6.3.3 | Hatch styles..... | 104 |
| 6.3.4 | Colour models..... | 104 |
| 6.3.5 | Prompt and echo types..... | 105 |
| 6.3.6 | Default parameter of gopen_gks | 105 |
| 7 | C GKS function interface..... | 106 |
| 7.1 | Notational conventions..... | 106 |
| 7.2 | Workstation independent functions..... | 106 |
| 7.2.1 | Control functions..... | 106 |
| 7.2.2 | Output functions..... | 108 |
| 7.2.3 | Design output functions..... | 110 |
| 7.2.4 | Primitive attribute functions..... | 113 |
| 7.2.5 | Normalization transformation functions..... | 119 |
| 7.2.6 | NDC picture functions..... | 120 |
| 7.2.7 | Metafile functions..... | 121 |
| 7.2.8 | Picture part store functions..... | 122 |
| 7.2.9 | Input functions..... | 124 |
| 7.2.10 | Font and glyph functions..... | 131 |
| 7.2.11 | Audit and playback functions..... | 131 |
| 7.2.12 | Inquiry functions..... | 132 |
| 7.2.13 | Utility functions..... | 145 |
| 7.3 | Workstation functions..... | 148 |
| 7.3.1 | Control functions..... | 148 |
| 7.3.2 | Inquiry functions..... | 155 |
| 7.3.3 | Retrieval functions..... | 172 |
| 7.3.4 | Viewing utility functions..... | 173 |
| 7.3.5 | Colour utility functions..... | 173 |
| 7.4 | Segment functions and workstation activation functions..... | 173 |
| 7.4.1 | Segment functions..... | 173 |
| 7.4.2 | Workstation activation functions..... | 176 |
| 7.4.3 | Utility functions..... | 176 |
| | Annexes..... | 177 |
| A | Compiled GKS/C specification..... | 177 |
| A.1 | Data types in compilation order..... | 177 |
| A.2 | Macros..... | 223 |
| A.3 | Function calls..... | 231 |
| A.4 | Compatibility layer..... | 260 |
| B | Sample programs..... | 271 |
| B.1 | STAR..... | 271 |
| B.2 | IRON..... | 273 |
| C | Short function identifiers..... | 280 |
| C.1 | In order of appearance..... | 280 |
| C.2 | In alphabetical order..... | 287 |
| D | Memory management..... | 294 |
| D.1 | Introduction..... | 294 |
| D.2 | Functions that return simple lists..... | 294 |
| D.2.1 | Operation of ginq_list_line_inds | 295 |
| D.3 | Functions that return structured data..... | 297 |
| D.3.1 | Operation of gcreate_store | 298 |

| | | | |
|---|-------|--|-----|
| | D.3.2 | Operation of ginq_stroke_st and ginq_pat_rep | 300 |
| | D.3.3 | Operation of gdel_store | 304 |
| E | | Compatibility with the 1991 edition of ISO/IEC 86514 | 307 |
| | E.1 | Comparison of this edition of ISO/IEC 86514 with the 1991 edition..... | 307 |
| | | E.1.1 Changes in ISO/IEC 86514 data types..... | 307 |
| | | E.1.2 Changes in ISO/IEC 86514 functions | 308 |
| | E.2 | The compatibility layer | 309 |
| | E.3 | The header gks_compat.h | 309 |
| | E.4 | Data types in gks_compat.h | 309 |
| | | E.4.1 Renamed data types | 309 |
| | | E.4.2 Renamed fields of data types | 309 |
| | | E.4.3 Obsolete data types..... | 310 |
| | E.5 | Macros | 314 |
| | E.6 | Functions in the compatibility layer..... | 314 |
| | | E.6.1 Replaced functions..... | 314 |
| | | E.6.2 Obsolete functions | 317 |
| F | | Function lists..... | 324 |
| | F.1 | Alphabetic by GKS name..... | 324 |
| | F.2 | Alphabetic by binding name | 331 |

iTeh STANDARD PREVIEW (standards.iteh.ai)

SIST EN ISO/IEC 8651-4:1997

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997>

Foreword

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

International Standard ISO/IEC 8651-4 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 24, *Computer graphics and image processing*.

This second edition cancels and replaces the first edition (ISO/IEC 8651-4:1991), which has been technically revised.

ISO/IEC 8651-4 consists of the following parts, under the general title *Information technology — Computer graphics — Graphical Kernel System (GKS)* language bindings.

- Part 1: FORTRAN
- Part 2: Pascal
- Part 3: Ada
- Part 4: C

Annexes A to F of this part of ISO/IEC 8651 are for information only.

Introduction

The Graphical Kernel System (GKS) functional description is registered as ISO/IEC 7942-1:1994. As explained in the Scope and Field of Application of ISO/IEC 7942-1, that International Standard is specified in a language independent manner and needs to be embedded in language dependent layers (language bindings) for use with particular programming languages.

The purpose of this part of ISO/IEC 8651 is to define a standard binding for the C computer programming language.

iTeh STANDARD PREVIEW (standards.iteh.ai)

[SIST EN ISO/IEC 8651-4:1997](https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997)

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997>

INTERNATIONAL STANDARD © ISO/IECISO/IEC 8651-4:1995(E)

Information technology — Computer graphics — Graphical Kernel System (GKS) language bindings —

Part 4:

C

1 Scope

iTeh STANDARD PREVIEW
(standards.iteh.ai)

The Graphical Kernel System (GKS), ISO/IEC 7942-1:1994, specifies a language independent nucleus of a graphics system. For integration into a programming language, GKS is embedded in a language dependent layer obeying the particular conventions of that language. This part of ISO/IEC 8651 specifies such a language dependent layer for the C language.

2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 8651. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 8651 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO/IEC 7942-1:1994, *Information technology – Computer graphics and image processing – Graphical Kernel System (GKS) – Part 1 : Functional description*.

ISO/IEC 9899:1990, *Programming languages – C*.

ISO/IEC TR 9973:1994, *Information technology – Computer graphics and image processing – Procedures for registration of graphical Items*.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

[SIST EN ISO/IEC 8651-4:1997](https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997)

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997>

3 The C language binding

The C language binding of GKS shall be as described in clauses 3 to 7.

3.1 Classification and designation

This part of ISO/IEC 8651 incorporates the rules of conformance defined in the GKS Standard (ISO/IEC 7942-1) for GKS implementations, with those additional requirements specifically defined for C bindings in GKS.

The following criteria shall determine conformance of an implementation to this part of ISO/IEC 8651:

In order to conform, an implementation of the C binding of GKS shall implement GKS as specified in ISO/IEC 7942-1. It shall make visible all of the declarations in the C binding specified in this part of ISO/IEC 8651 for a specific level of C.

Thus, for example, the syntax of the function names shall be precisely as specified in the binding and parameters shall be of the data types stated in the binding.

3.2 Functions versus macros

An implementation may substitute macros for functions. However, the macros shall be designed so that side-effects work properly. In general, a macro cannot be used to replace the error handling function **gerr_hand**. See also 3.10.

3.3 Character strings

The C language represents character strings as an array of characters terminated by the null character (i.e. '\0'). This means that the null character is not usable as a printable character.

3.4 Function identifiers

The function names of GKS are all mapped to C functions which begin with the letter **g**. Words and phrases used in the GKS function names are often abbreviated in the representation and are always separated with the underscore character **"_"**. The set of such abbreviations is given in 4.2, and the resulting C function names are listed in 4.3. For example, the abbreviation for the GKS function DELETE SEGMENT FROM WORKSTATION is **gdel_seg_ws**. **del**, **seg**, and **ws** are abbreviations for DELETE, SEGMENT, and WORKSTATION. The conjunctive FROM is mapped to the null string.

The C language (ISO/IEC 9899) requires that compilers recognize internal identifiers which are distinct in at least 31 characters. That standard also requires that external identifiers (i.e. those seen by the linker) be recognized to a minimum of six characters, independent of case.

Implementations which run in environments where two distinct C internal identifiers would be equivalent, if they were both external identifiers, shall include a set of object-like macros in the header which equate the long names to a set of short names. A possible set of short names for a compiler that accepts only 8 characters for external definitions may be found in annex C.

3.5 Registration

ISO/IEC 7942 reserves certain value ranges for registration¹ as graphical items. The registered graphical items will be bound to the C programming language (and other programming languages). The registered item binding will be consistent with the binding presented in this part of ISO/IEC 8651.

¹ For the purpose of this part of ISO/IEC 8651 and according to the rules for the designation and operation of registration authorities in the ISO/IEC Directives, the ISO/IEC council has designated the National Institute of Standards and Technology (Institute of Computer Sciences and Technology), A-266 Technology Building, Gaithersburg, MD 20899, USA to act as registration authority.

3.6 Identifiers for graphical items

Generalized Drawing Primitives and Escape functions are referenced via identifiers. This part of ISO/IEC 8651 specifies the format of the identifiers but it does not specify the registration of the identifiers. The identifiers are used as arguments to the functions **ggdp** and **gescape**.

An implementation may also represent GDPs and Escapes as separate functions, but this is not required.

There are two formats for these identifiers. One format is for registered GDPs and Escapes and the other format is for unregistered GDPs and Escapes.

The format for registered GDP identifiers is:

```
#define      GGDP_Rn      (n)          /* where 'n' is the registered GDP
                                         identifier */
```

The format for unregistered GDP identifiers is:

```
#define      GGDP_Un      (-n)         /* where 'n' is implementation
                                         dependent */
```

The format for registered Escape function identifiers is:

```
#define      GESCAPE_Rn   (n)          /* where 'n' is the registered
                                         Escape identifier */
```

The format for unregistered Escape function identifiers is:

```
#define      GESCAPE_Un   (-n)         /* where 'n' is implementation
                                         dependent */
```

iTeh STANDARD PREVIEW
(standards.iteh.ai)

3.7 Return values

All GKS/C functions have return value **void**.

3.8 Headers

[SIST EN ISO/IEC 8651-4:1997](https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997)

<https://standards.iteh.ai/catalog/standards/sist/e20e87cb-e79b-41b8-9c94-e283b4ced76d/sist-en-iso-iec-8651-4-1997>

3.8.1 **gks.h**

C provides a mechanism to access information stored in a header via the **#include** preprocessing directive. Clause 5 of this part of ISO/IEC 8651 describes the data types that shall be defined in the header **gks.h** which shall be included in any application program that intends to use GKS via the C binding.

This part of ISO/IEC 8651 uses the data type **size_t** (inter alia as a field in the data type **Gdata**). The type **size_t** is environment-defined (i.e. **unsigned long** or **unsigned int**) and is defined in the headers **<stdio.h>**, **<stddef.h>**, **<stdlib.h>**, **<string.h>**, **<time.h>**.

Additional implementation-dependent items may be placed in this header if needed. These items should start with the sentinel "G" or "g", as far as applicable.

The header **gks.h** shall also contain external declarations for all GKS/C functions because they have a **void** return type. For example, the declaration for the function **gopen_gks** would look like this:

```
extern void gopen_gks(const char *err_file, size_t memory);
```

3.8.2 **gks_compat.h**

For application programs which used to run on top of the 1991 edition of this part of ISO/IEC, the header **gks_compat.h** is provided. **gks_compat.h** includes GKS/C data types that are no longer supported, as well as external declarations for all GKS/C functions that are no longer supported. Implementations of this part of ISO/IEC 8651 shall support these functions in a compatibility layer, according to the guidelines in Annex G of ISO/IEC 7942-1:1994.

3.9 Memory management

The application shall allocate the memory needed for the data returned by the implementation. In general, the application will allocate a C structure and pass a pointer to that structure to an inquiry routine, which will then place information into the structure. However, a number of inquiry functions return variable length data, the length of which is not known *a priori* by the application.

These functions fall into two classes. One class of functions returns a simple, homogeneous, list of items. For example, the function INQUIRE LIST OF MARKER INDICES returns a list of the available marker indices. The other class returns complex, heterogeneous data structures. For example, the function INQUIRE GKS STATE LIST ENTRY returns a piece of the GKS state which may include several data structures of different length. The binding of these two classes of functions is described in detail below. Subclause 3.10 describes the errors that can be invoked during execution of functions which use the memory management policy.

3.9.1 Functions which return simple lists

Inquiry functions which return a list of items are bound such that the application can inquire about a portion of the list. This list is a subset of the implementation's internal list and is called the application's list. This allows the application to process the implementation's list in a piecewise manner rather than all at once.

The application allocates the memory for a list and passes that list to the implementation. The implementation places the results of the inquiry into the list. In order to support this policy of memory management, three additional parameters have been added to functions which return lists:

- a) **num_elems_appl_list**: An integer input parameter which is the length of the application's list. The value of **num_elems_appl_list** indicates the number of items (i.e. list elements) which will fit into the application list. A value of 0 is valid and allows the application to determine the size of the implementation's list (which is returned via **num_elems_impl_list**) without having the implementation return any of the elements of its list. If **num_elems_appl_list** is negative, **GE_APPL_LIST_LENGTH_LT_ZERO** is returned as the value of the error indicator parameter.
- b) **start_ind**: An integer input parameter which is an index into the implementation's list. (Index 0 is the first element of both the implementation's and application's list.) **start_ind** indicates the first item in the implementation's list that is copied into index 0 of the application's list. Items are copied sequentially from the implementation's list into the application's list until the application's list is full or there are no more items in the implementation's list. If **start_ind** is out of range, error **GE_START_IND_INVAL** is returned as the value of the error indicator parameter.
- c) **num_elems_impl_list**: An output parameter which is a pointer to an integer. The implementation stores into this parameter the number of items that are in the implementation's list.

In annex D, a possible underlying mechanism is described.

3.9.2 Functions which return complex data structures

The data returned by inter alia the ESCAPE function, the AWAIT INPUT function and the functions which return state lists or description tables can be complex in structure. They cannot be represented by a simple list of items. It would be an onerous task for the application to have to allocate and prepare data structures for these routines. In order to facilitate this task of using these inquiry functions, the binding defines a new resource, called a *Store*, to manage the memory for these functions.

The *Store* resource is opaque to the application. The application does not know the structure of the *Store* or how it is implemented. The *Store* is defined as a **void ***. This part of ISO/IEC 8651 defines two new functions which create (in CREATE STORE, bound as **gcreate_store**) and delete (in DELETE STORE, bound as **del_store**) a *Store*.

A *Store* is used by the implementation to manage the memory needed by the functions which return complex data structures. Without specifying an implementation of a *Store*, it is safe to say that it will contain