

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

Part 3: Ada

The European Standard EN 28651-3:1992 has the status of a
British Standard

UDC 681.3.04:800.92

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW



Document Sharing Hub

Committees responsible for this British Standard

The preparation of this British Standard was entrusted by the Information Systems Technology Standards Committee (IST/-) to Technical Committee IST/31, upon which the following bodies were represented:

- British Computer Society
- British Office Technology Manufacturers' Alliance
- Business Equipment and Information Technology Association
- Computer Graphics Suppliers Association
- Middlesex Polytechnic Faculty of Art and Design
- National Computing Centre Ltd.
- Science and Engineering Research Council (Rutherford Appleton Laboratory)
- University of Leeds
- University of Leicester
- University of London Computer Centre
- University of Manchester

This British Standard, having been prepared under the direction of the Information Systems Technology Standards Policy Committee, was published under the authority of the Board of BSI and comes into effect on 31 August 1989

© BSI 05-2000

The following BSI references relate to the work on this standard:
Committee reference IST/31
Draft for comment DD 114-3

ISBN 0 580 17486 7

Amendments issued since publication

Amd. No.	Date of issue	Comments
7539	December 1992	Indicated by a sideline in the margin

Contents

	Page
Committees responsible	Inside front cover
National foreword	ii
Foreword	2
Text of EN 28651-3	3
Publications referred to	Inside back cover

National foreword

This Part of BS 7040 has been prepared under the direction of the Information Systems Technology Standards Policy Committee. It is identical with ISO 8651-3:1988 “*Information processing systems — Computer graphics — Graphical Kernel System (GKS) language binding — Part 3: Ada*” published by the International Organization for Standardization (ISO).

In 1992 the European Committee for Standardization (CEN) accepted ISO 8651-3:1988 as European Standard EN 28651-3:1992. As a consequence of implementing the European Standard this British Standard is renumbered as BS EN 28651-3 and any reference to BS 7040-3 should be read as a reference to BS EN 28651-3.

This Part of BS 7040 supersedes DD 114-3:1985, which is withdrawn.

Cross-references

International Standard	Corresponding British Standard
ISO 7942:1985	BS 6390:1985 <i>Specification for a set of functions for computer graphics programming, the Graphical Kernel System (GKS)</i> (Identical)

The Technical Committee has reviewed the provisions of ISO 8652, to which reference is made in the text, and has decided that they are acceptable for use in conjunction with this standard. ISO 8652:1987 is an endorsement by ISO of ANSI standard, ANSI/MIL-STD-1815A:1983.

Up-to-date information on existing and projected British Standards publications for Computer Graphics is included in the “BASIC LIST” published as part of the BSI Information Technology Services (BITS) initiative.

For details contact:

Marketing Department
BSI
Linford Wood
Milton Keynes
MK14 6LE

A British Standard does not purport to include all the necessary provisions of a contract. Users of British Standards are responsible for their correct application.

Compliance with a British Standard does not of itself confer immunity from legal obligations.

Summary of pages

This document comprises a front cover, an inside front cover, pages i and ii, the EN title page, page 2 to 188, an inside back cover and a back cover.

This standard has been updated (see copyright date) and may have had amendments incorporated. This will be indicated in the amendment table on the inside front cover.

UDC 681.3.04:800.92

Descriptors: Data processing, graphic data processing, computer interfaces, graphical kernel system, programming languages, Ada

English version

Information processing systems — Computer graphics — Graphical Kernel System (GKS) language bindings Part 3. Ada

(ISO 8651-3:1988)

Systèmes de traitement de l'information —
Infographie — Interfaces langage avec GKS
Partie 3. Ada
(ISO 8651-3:1988)

Graphische Systeme der
Informationsverarbeitung — Sprachbindungen
für das Graphische Kernsystem (GKS)
Teil 3. Ada
(ISO 8651-3:1988)

This European Standard was approved by CEN on 1992-08-13. 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.

This European Standard exists 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

© 1992 Copyright reserved to CEN members

Foreword

The Technical Board has decided to submit the International Standard 8651-3:1988 *Information processing systems — Computer graphics — Graphical Kernel System (GKS) language bindings — Part 3: Ada* for Formal Vote. The standard was accepted.

At present the standard exists in the English and French versions only.

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 February 1993 and conflicting national standards shall be withdrawn at the latest by February 1993.

According to the CEN/CENELEC Common Rules the following countries are bound to implement this standard:

Austria, Belgium, Denmark, Finland, France, Germany, Greece, Iceland, Ireland, Italy, Luxembourg, Netherlands, Norway, Portugal, Spain, Sweden, Switzerland, United Kingdom.

Contents

	Page
Foreword	2
0 Introduction	3
1 Scope and field of application	4
2 References	5
3 The Ada language binding of GKS	6
4 Tables	10
5 Functions in the Ada Binding to GKS	55
Annex A Compiled GKS Specification	100
Annex B Cross Reference Listing of Implementation Defined Items	150
Annex C Example Programs	151
Annex D GKS Multi-Tasking	169
Annex E Unsupported generalized drawing primitives and escapes	174
Annex F Metafile Item Types	177
Annex G Index of GKS Functions	179
Table 1 — Abbreviations used in Procedure names	10
Table 2 — List of procedures using the abbreviations	10
Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name	14
Table 4 — Alphabetical list of GKS functions by level	19

0 Introduction

The Graphical Kernel System (GKS) (ISO 7942) 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 8651 is to define a standard binding for the Ada computer programming language.

1 Scope and field of application

ISO 7942 (GKS) 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 8651 specifies such a language dependent layer for the Ada language.

2 References

ISO 7942, *Information processing systems — Computer graphics — Graphical Kernel System (GKS) functional description*.

ISO 8652, *Programming Languages — Ada*.

3 The Ada language binding of GKS

This binding does not assume that the compiler supports any Ada language features which are implementation dependent, but implies that the compiler shall be able to support the declarations contained in this GKS/Ada binding. This binding does not make any assumptions regarding the machine representation of the predefined Ada numeric types.

This binding assumes that the application programmer will supply an error file name and connection identifier that are in an acceptable format for the Ada implementation.

This binding makes no assumptions regarding the format of a string specifying an error file name or connection identifier for devices or metafiles.

3.1 Conformance

This binding incorporates the rules of conformance defined in the GKS Standard (ISO 7942) for GKS implementations, with these additional requirements specifically defined for Ada implementations of GKS.

The following criteria are established for determining conformance or non-conformance of an implementation to this binding:

- a) An implementation of GKS in Ada conforms to a level of GKS if it makes visible exactly the declarations for that level of GKS and lower levels of GKS as stated in this binding.
- b) The semantics of an implementation shall be those stated in the GKS standard as modified or extended for Ada as stated in this binding document.
- c) The package corresponding to the GKS level being implemented shall be an available Ada library unit, with all names as specified by this document.

3.2 Implications of the language

3.2.1 Functional mapping

The functions of GKS are all mapped to Ada procedures. The mapping utilizes a one-to-one correspondence between the GKS functions and Ada procedures, except for the GKS functions Inquire Current Primitive Attribute Values and Inquire Current Individual Attribute Values. These are bound with one Ada procedure for each of the attributes being inquired; in addition, the attributes are bound as a single record.

3.2.2 Implementation and host dependencies

There are a number of implementation and host dependencies associated with the Ada compiler and runtime system used. These will affect the portability of application programs and their use of GKS. The application programmer should follow accepted practices for ensuring portability of Ada programs to avoid introducing problems when rehosting the application on another system. Implementation dependencies include runtime storage management and processor management.

3.2.3 Error handling

The inquiry functions utilize error indicator parameters for the error returns, and do not raise Ada exceptions. The application program must ensure that these error indicators are checked before attempting to access other parameters, since some Ada implementations do not raise an exception if an undefined value is accessed.

The error handling requirements of GKS can be summarized as follows:

1. By default, a procedure named `ERROR_HANDLING` will be provided that simply reports the error by calling `ERROR_LOGGING`. This is called from the GKS function that detects the error.
2. The `ERROR_HANDLING` procedure may be replaced by one defined by the user.

The procedure `ERROR_HANDLING` is defined as a library subprogram:

```
with GKS_TYPES;
use GKS_TYPES;
procedure ERROR_HANDLING (ERROR_INDICATOR : in ERROR_NUMBER;
                           GKS_FUNCTION   : in STRING;
                           ERROR_FILE     : in STRING
                           := DEFAULT_ERROR_FILE);

-- The procedure ERROR_HANDLING is defined as a library subprogram, and is not
-- declared within package GKS.
```

This binding defines two different bodies for this subprogram; each must be supplied by the implementation. The default body is the one required by GKS semantics. It simply calls `ERROR_LOGGING` and returns. The second body calls `ERROR_LOGGING` and then raises the exception `GKS_ERROR`. The GKS function must be written so as not to handle `GKS_ERROR` (this is a requirement of the implementation). Thus, by Ada rules, the exception will be propagated back to the application program that called the GKS function in which the error was detected.

The means by which the user replaces the default body of either the exception-raising version or another one of his or her choosing is dependent upon the Ada library manager. Some implementations support multiple versions of a body with a single specification or otherwise allow hierarchical libraries with the sharing of common units. In other implementations it may be necessary to duplicate the GKS library for each version of `ERROR_HANDLING`.

GKS errors are mapped to the single exception `GKS_ERROR`, declared in the GKS package. The expected style in dealing with errors using exception handling is to provide a handler for the `GKS_ERROR` exception.

3.2.4 Data mapping

The simple and compound data types of GKS are bound to a variety of Ada scalar and compound types. Constraints on permitted values are reflected where possible in the type definitions. The general correspondence between the GKS data types and Ada binding data types is summarized below:

GKS integers are mapped to Ada integer types.

GKS reals are mapped to Ada floating-point types.

GKS strings are mapped to the predefined Ada type `STRING`, or to a type providing for variable length strings.

GKS points are mapped to Ada record types.

GKS names are mapped to Ada discrete types.

GKS enumeration types are mapped to Ada enumeration types.

GKS vectors are mapped to Ada record types.

GKS matrices are mapped to Ada array types.

GKS lists (of elements of a particular type) are mapped to an Ada private type declared in an instantiation of the generic GKS_LIST_UTILITIES package.

GKS arrays are mapped to either an unconstrained Ada array type, or to a record type providing for variable length arrays.

GKS ordered pairs are mapped to Ada record types.

GKS data records are mapped to Ada private types. In some cases a set of subprograms for operating on the data records are explicitly defined by this binding. This is because the content and structure of the data record is implementation-dependent. An implementation of GKS may provide other subprograms for manipulating implementation-dependent data records.

3.2.5 Multi-tasking

The Ada language definition provides explicit support for concurrency. The Ada tasking model includes facilities for declaring and allocating tasks, and operations allowing intertask communication and synchronization.

The GKS standard, and hence this binding, neither requires nor prohibits an implementation from protecting against problems which could arise from asynchronous access to the GKS data structures from concurrent tasks. Implementors of GKS should provide information in the user's documentation regarding whether protection against such problems is implemented.

Annex D contains guidelines for implementors who want to support multi-tasking application programs. This annex does not form an integral part of the binding standard, but provides additional information.

3.2.6 Packaging

The GKS standard defines nine levels of graphic functionality, with level 0a as the lowest level and level 2c as the highest level. An implementation of GKS may implement every level individually or as a single system. To support this concept this binding defines nine Ada packages which correspond to each of the GKS levels. Each of these packages is named

package GKS is ... end GKS;

to provide portability of application programs for levels of GKS. However, the contents of the packages differ depending on the level of GKS that they provide. Each of these packages provides the subprograms defined for its level and all subprograms defined in "lower" levels as described in 5.1 of this binding.

Associated with each of these packages is a data type package which provides the type declarations for the appropriate level as defined in 4.2 and the GKS defined exception defined in 4.3.1. These packages are named

package GKS_TYPES is ... end GKS_TYPES;

The Ada program library facility should be used to provide the levels separation. Thus, an Ada graphics application program which uses GKS would "with" the appropriate GKS packages which provide the subprogram, types, and exceptions for that level by compiling and linking to the corresponding Ada library which contains that level of GKS. For example, an application which uses level 0a would "with" the packages as follows:

```
with GKS;
use GKS_TYPES;
procedure APPLICATION is
begin
  null;
end APPLICATION;
```

Then the program is compiled and linked to the Ada program library that corresponds to level 0a. Several additional Ada units are defined in this binding. These are:

- o generic package GKS_COORDINATE_SYSTEM
- o generic package GKS_LIST_UTILITIES

These generic packages support the declaration types in the GKS_TYPES package described above. The GKS_COORDINATE_SYSTEM is a generic package that defines an assortment of types for supporting each of the GKS coordinate systems. GKS_LIST_UTILITIES is also a generic package which provides type declarations and operations for list types which correspond to the GKS list types.

3.2.7 *Application program environment*

An application program utilizing an Ada implementation of GKS will need to be aware of the environment in which both GKS and the application program(s) reside.

One such interface is the Ada program library. The Ada language requires that the application program have access to the program library in which the GKS software resides. The ISO 8652 Ada Reference Manual does not specify whether there is a single library or multiple libraries, or how access to the libraries is granted, managed, etc. The user's documentation for the GKS implementation should specify where the GKS library exists in the system, and how access to the library is acquired.

Input/Output interfaces are also implementation-dependent, and are required to be described in the user's documentation. Besides the obvious graphics device interface information, interfaces to the file system shall be included in the documentation. Specifically, this includes the interface to the GKS error file and also the metafile storage.

3.2.8 *Registration¹⁾*

The GKS standard reserves certain value ranges for registration as graphical items. The registered graphical items will be bound to Ada (and other programming languages). The registered item bindings will be consistent with the binding presented in the document.

¹⁾ For the purpose of this part of ISO 8651 and according to the rules for the designation and operation of registration authorities in the ISO Directives, the ISO Council has designated the National Bureau of Standards (Institute of Computer Sciences and Technology) A266 Technology Building, Gaithersburg, MD, 20899, USA to act as registration authority.

4 Tables

4.1 Procedures

Table 1 — Abbreviations used in Procedure names

ASF	aspect source flag
CHAR	character
ESC	escape
GDP	generalized drawing primitive
GKS	Graphical Kernel System
GKSM	Graphical Kernel System metafile
ID	identifier
INQ	inquire
MAX	maximum
UGDP	unregistered generalized drawing primitive
UESC	unregistered escape
WS	workstation(s)

Table 2 — List of procedures using the abbreviations

ASF	INQ_LIST_OF_ASF SET_ASF
CHAR	INQ_CHAR_BASE_VECTOR INQ_CHAR_EXPANSION_FACTOR INQ_CHAR_HEIGHT INQ_CHAR_WIDTH INQ_CHAR_SPACING INQ_CHAR_UP_VECTOR SET_CHAR_EXPANSION_FACTOR SET_CHAR_HEIGHT SET_CHAR_SPACING SET_CHAR_UP_VECTOR
ESC	ESC UESC
GDP	GDP INQ_GDP INQ_LIST_OF_AVAILABLE_GDP UGDP
GKS	CLOSE_GKS EMERGENCY_CLOSE_GKS INQ_LEVEL_OF_GKS OPEN_GKS

Table 2 — List of procedures using the abbreviations

GKSM

GET_ITEM_TYPE_FROM_GKSM
 READ_ITEM_FROM_GKSM
 WRITE_ITEM_TO_GKSM

ID

INQ_CURRENT_PICK_ID_VALUE
 SET_PICK_ID

INQ

INQ_CHAR_BASE_VECTOR
 INQ_CHAR_EXPANSION_FACTOR
 INQ_CHAR_HEIGHT
 INQ_CHAR_WIDTH
 INQ_CHAR_SPACING
 INQ_CHAR_UP_VECTOR
 INQ_CHOICE_DEVICE_STATE
 INQ_CLIPPING
 INQ_COLOUR_FACILITIES
 INQ_COLOUR_REPRESENTATION
 INQ_CURRENT_NORMALIZATION_TRANSFORMATION_NUMBER
 INQ_CURRENT_INDIVIDUAL_ATTRIBUTE_VALUES
 INQ_CURRENT_PICK_ID_VALUE
 INQ_CURRENT_PRIMITIVE_ATTRIBUTE_VALUES
 INQ_DEFAULT_CHOICE_DEVICE_DATA
 INQ_DEFAULT_DEFERRAL_STATE_VALUES
 INQ_DEFAULT_LOCATOR_DEVICE_DATA
 INQ_DEFAULT_PICK_DEVICE_DATA
 INQ_DEFAULT_STRING_DEVICE_DATA
 INQ_DEFAULT_STROKE_DEVICE_DATA
 INQ_DEFAULT_VALUATOR_DEVICE_DATA
 INQ_DISPLAY_SPACE_SIZE
 INQ_DYNAMIC_MODIFICATION_OF_SEGMENT_ATTRIBUTES
 INQ_DYNAMIC_MODIFICATION_OF_WS_ATTRIBUTES
 INQ_FILL_AREA_COLOUR_INDEX
 INQ_FILL_AREA_FACILITIES
 INQ_FILL_AREA_INDEX
 INQ_FILL_AREA_INTERIOR_STYLE
 INQ_FILL_AREA_REPRESENTATION
 INQ_FILL_AREA_STYLE_INDEX
 INQ_GDP
 INQ_INPUT_QUEUE_OVERFLOW
 INQ_LEVEL_OF_GKS
 INQ_LIST_OF_ASF
 INQ_LINETYPE
 INQ_LINEWIDTH_SCALE_FACTOR
 INQ_LIST_OF_AVAILABLE_GDP
 INQ_LIST_OF_AVAILABLE_WS_TYPE
 INQ_LIST_OF_COLOUR_INDICES
 INQ_LIST_OF_FILL_AREA_INDICES

Table 2 — List of procedures using the abbreviations

INQ_LIST_OF_NORMALIZATION_TRANSFORMATION_NUMBER
 INQ_LIST_OF_PATTERN_INDICES
 INQ_LIST_OF_POLYLINE_INDICES
 INQ_LIST_OF_POLYMARKER_INDICES
 INQ_LIST_OF_TEXT_INDICES
 INQ_LOCATOR_DEVICE_STATE
 INQ_MAX_LENGTH_OF_WS_STATE_TABLES
 INQ_MAX_NORMALIZATION_TRANSFORMATION_NUMBER
 INQ_MORE_SIMULTANEOUS_EVENTS
 INQ_NAME_OF_OPEN_SEGMENT
 INQ_NORMALIZATION_TRANSFORMATION
 INQ_NUMBER_OF_SEGMENT_PRIORITIES_SUPPORTED
 INQ_NUMBER_OF_AVAILABLE_LOGICAL_INPUT_DEVICES
 INQ_OPERATING_STATE_VALUE
 INQ_PATTERN_FACILITIES
 INQ_PATTERN_HEIGHT_VECTOR
 INQ_PATTERN_REFERENCE_POINT
 INQ_PATTERN_REPRESENTATION
 INQ_PATTERN_WIDTH VECTOR
 INQ_PICK_DEVICE_STATE
 INQ_PIXEL
 INQ_PIXEL_ARRAY
 INQ_PIXEL_ARRAY_DIMENSIONS
 INQ_POLYLINE_COLOUR_INDEX
 INQ_POLYLINE_FACILITIES
 INQ_POLYLINE_INDEX
 INQ_POLYLINE_REPRESENTATION
 INQ_POLYMARKER_REPRESENTATION
 INQ_POLYMARKER_COLOUR_INDEX
 INQ_POLYMARKER_INDEX
 INQ_POLYMARKER_FACILITIES
 INQ_POLYMARKER_SIZE_SCALE_FACTOR
 INQ_POLYMARKER_TYPE
 INQ_PREDEFINED_COLOUR_REPRESENTATION
 INQ_PREDEFINED_FILL_AREA_REPRESENTATION
 INQ_PREDEFINED_PATTERN_REPRESENTATION
 INQ_PREDEFINED_POLYLINE_REPRESENTATION
 INQ_PREDEFINED_POLYMARKER_REPRESENTATION
 INQ_PREDEFINED_TEXT_REPRESENTATION
 INQ_SEGMENT_ATTRIBUTES
 INQ_SET_OF_ACTIVE_WS
 INQ_SET_OF_ASSOCIATED_WS
 INQ_SET_OF_OPEN_WS
 INQ_SET_OF_SEGMENT_NAMES_IN_USE
 INQ_SET_OF_SEGMENT_NAMES_ON_WS
 INQ_STRING_DEVICE_STATE

Table 2 — List of procedures using the abbreviations

	INQ_STROKE_DEVICE_STATE
	INQ_TEXT_ALIGNMENT
	INQ_TEXT_COLOUR_INDEX
	INQ_TEXT_EXTENT
	INQ_TEXT_FACILITIES
	INQ_TEXT_FONT_AND_PRECISION
	INQ_TEXT_INDEX
	INQ_TEXT_PATH
	INQ_TEXT_REPRESENTATION
	INQ_VALUATOR_DEVICE_STATE
	INQ_WS_CATEGORY
	INQ_WS_CLASSIFICATION
	INQ_WS_CONNECTION_AND_TYPE
	INQ_WS_DEFERRAL_AND_UPDATE_STATES
	INQ_WS_MAX_NUMBER
	INQ_WS_STATE
	INQ_WS_TRANSFORMATION
MAX	
	INQ_MAX_LENGTH_OF_WS_STATE_TABLES
	INQ_MAX_NORMALIZATION_TRANSFORMATION_NUMBER
	INQ_WS_MAX_NUMBERS
WS	
	ACTIVATE_WS
	ASSOCIATE_SEGMENT_WITH_WS
	CLEAR_WS
	CLOSE_WS
	COPY_SEGMENT_TO_WS
	DEACTIVATE_WS
	DELETE_SEGMENT_FROM_WS
	INQ_DYNAMIC_MODIFICATION_OF_WS_ATTRIBUTES
	INQ_LIST_OF_AVAILABLE_WS_TYPE
	INQ_MAX_LENGTH_OF_WS_STATE_TABLES
	INQ_SET_OF_ACTIVE_WS
	INQ_SET_OF_ASSOCIATED_WS
	INQ_SET_OF_OPEN_WS
	INQ_SET_OF_SEGMENT_NAMES_ON_WS
	INQ_WS_CATEGORY
	INQ_WS_CLASSIFICATION
	INQ_WS_CONNECTION_AND_TYPE
	INQ_WS_DEFERRAL_AND_UPDATE_STATES
	INQ_WS_MAX_NUMBER
	INQ_WS_STATE
	INQ_WS_TRANSFORMATION
	OPEN_WS
	REDRAW_ALL_SEGMENTS_ON_WS
	SET_WS_VIEWPORT
	SET_WS_WINDOW
	UPDATE_WS

Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name

<u>Ada Bound Name</u>	<u>GKS Function</u>
ACCUMULATE_TRANSFORMATION_MATRIX	ACCUMULATE TRANSFORMATION MATRIX
ACTIVATE_WS	ACTIVATE WORKSTATION
ASSOCIATE_SEGMENT_WITH_WS	ASSOCIATE SEGMENT WITH WORKSTATION
AWAIT_EVENT	AWAIT EVENT
CELL_ARRAY	CELL ARRAY
CLEAR_WS	CLEAR WORKSTATION
CLOSE_GKS	CLOSE GKS
CLOSE_SEGMENT	CLOSE SEGMENT
CLOSE_WS	CLOSE WORKSTATION
COPY_SEGMENT_TO_WS	COPY SEGMENT TO WORKSTATION
CREATE_SEGMENT	CREATE SEGMENT
DEACTIVATE_WS	DEACTIVATE WORKSTATION
DELETE_SEGMENT	DELETE SEGMENT
DELETE_SEGMENT_FROM_WS	DELETE SEGMENT FROM WORKSTATION
EMERGENCY_CLOSE_GKS	EMERGENCY CLOSE GKS
ERROR_HANDLING	ERROR HANDLING
ERROR_LOGGING	ERROR LOGGING
ESCAPE	ESCAPE
EVALUATE_TRANSFORMATION_MATRIX	EVALUATE TRANSFORMATION MATRIX
FILL_AREA	FILL AREA
FLUSH_DEVICE_EVENTS	FLUSH DEVICE EVENTS
GDP	GENERALIZED DRAWING PRIMITIVE
GET_CHOICE	GET CHOICE
GET_ITEM_TYPE_FROM_GKSM	GET ITEM TYPE FROM GKSM
GET_LOCATOR	GET LOCATOR
GET_PICK	GET PICK
GET_STRING	GET STRING
GET_STROKE	GET STROKE
GET_VALUATOR	GET VALUATOR
INITIALISE_CHOICE	INITIALISE CHOICE
INITIALISE_LOCATOR	INITIALISE LOCATOR
INITIALISE_PICK	INITIALISE PICK
INITIALISE_STRING	INITIALISE STRING
INITIALISE_STROKE	INITIALISE STROKE
INITIALISE_VALUATOR	INITIALISE VALUATOR
INQ_CHOICE_DEVICE_STATE	INQUIRE CHOICE DEVICE STATE
INQ_CLIPPING	INQUIRE CLIPPING
INQ_COLOUR_FACILITIES	INQUIRE COLOUR FACILITIES
INQ_COLOUR_REPRESENTATION	INQUIRE COLOUR REPRESENTATION

Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name

<u>Ada Bound Name</u>	<u>GKS Function</u>
INQ_CURRENT_INDIVIDUAL_ATTRIBUTE_VALUES	INQUIRE CURRENT INDIVIDUAL ATTRIBUTE VALUES
The following functions are a one-to-many mapping of the GKS function “inquire current individual attribute values”.	
INQ_CHAR_EXPANSION_FACTOR	
INQ_CHAR_SPACING	
INQ_FILL_AREA_COLOUR_INDEX	
INQ_FILL_AREA_INTERIOR_STYLE	
INQ_FILL_AREA_STYLE_INDEX	
INQ_LINETYPE	
INQ_LINEWIDTH_SCALE_FACTOR	
INQ_LIST_OF_ASF	
INQ_POLYLINE_COLOUR_INDEX	
INQ_POLYMARKER_COLOUR_INDEX	
INQ_POLYMARKER_SIZE_SCALE_FACTOR	
INQ_POLYMARKER_TYPE	
INQ_TEXT_COLOUR_INDEX	
INQ_TEXT_FONT_AND_PRECISION	
INQ_CURRENT_NORMALIZATION_ TRANSFORMATION_NUMBER	INQUIRE CURRENT NORMALIZATION TRANSFORMATION NUMBER
INQ_CURRENT_PICK_ID_VALUE	INQUIRE CURRENT PICK IDENTIFIER VALUE
INQ_CURRENT_PRIMITIVE_ ATTRIBUTE_VALUES	INQUIRE CURRENT PRIMITIVE ATTRIBUTE VALUES
The following functions are a one-to-many mapping of the GKS function “inquire current primitive attribute values”.	
INQ_CHAR_BASE_VECTOR	
INQ_CHAR_HEIGHT	
INQ_CHAR_WIDTH	
INQ_CHAR_UP_VECTOR	
INQ_FILL_AREA_INDEX	
INQ_PATTERN_HEIGHT_VECTOR	
INQ_PATTERN_REFERENCE_POINT	
INQ_PATTERN_WIDTH_VECTOR	
INQ_POLYLINE_INDEX	
INQ_POLYMARKER_INDEX	
INQ_TEXT_ALIGNMENT	
INQ_TEXT_INDEX	
INQ_TEXT_PATH	
INQ_DEFAULT_CHOICE_DEVICE_DATA	INQUIRE DEFAULT CHOICE DEVICE DATA
INQ_DEFAULT_DEFERRAL_STATE_VALUES	INQUIRE DEFAULT DEFERRAL STATE VALUES
INQ_DEFAULT_LOCATOR_DEVICE_DATA	INQUIRE DEFAULT LOCATOR DEVICE DATA
INQ_DEFAULT_PICK_DEVICE_DATA	INQUIRE DEFAULT PICK DEVICE DATA

Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name

<u>Ada Bound Name</u>	<u>GKS Function</u>
INQ_DEFAULT_STRING_DEVICE_DATA	INQUIRE DEFAULT STRING DEVICE DATA
INQ_DEFAULT_STROKE_DEVICE_DATA	INQUIRE DEFAULT STROKE DEVICE DATA
INQ_DEFAULT_VALUATOR_DEVICE_DATA	INQUIRE DEFAULT VALUATOR DEVICE DATA
INQ_DISPLAY_SPACE_SIZE	INQUIRE DISPLAY SPACE SIZE
INQ_DYNAMIC_MODIFICATION_OF_SEGMENT_ATTRIBUTES	INQUIRE DYNAMIC MODIFICATION OF SEGMENT ATTRIBUTES
INQ_DYNAMIC_MODIFICATION_OF_WS_ATTRIBUTES	INQUIRE DYNAMIC MODIFICATION OF WORKSTATION ATTRIBUTES
INQ_FILL_AREA_FACILITIES	INQUIRE FILL AREA FACILITIES
INQ_FILL_AREA_REPRESENTATION	INQUIRE FILL AREA REPRESENTATION
INQ_GDP	INQUIRE GENERALIZED DRAWING PRIMITIVE
INQ_INPUT_QUEUE_OVERFLOW	INQUIRE INPUT QUEUE OVERFLOW
INQ_LEVEL_OF_GKS	INQUIRE LEVEL OF GKS
INQ_LIST_OF_AVAILABLE_GDP	INQUIRE LIST OF AVAILABLE GENERALIZED DRAWING PRIMITIVES
INQ_LIST_OF_AVAILABLE_WS_TYPES	INQUIRE LIST OF AVAILABLE WORKSTATION TYPES
INQ_LIST_OF_COLOUR_INDICES	INQUIRE LIST OF COLOUR INDICES
INQ_LIST_OF_FILL_AREA_INDICES	INQUIRE LIST OF FILL AREA INDICES
INQ_LIST_OF_NORMALIZATION_TRANSFORMATION_NUMBERS	INQUIRE LIST OF NORMALIZATION TRANSFORMATION NUMBERS
INQ_LIST_OF_PATTERN_INDICES	INQUIRE LIST OF PATTERN INDICES
INQ_LIST_OF_POLYLINE_INDICES	INQUIRE LIST OF POLYLINE INDICES
INQ_LIST_OF_POLYMARKER_INDICES	INQUIRE LIST OF POLYMARKER INDICES
INQ_LIST_OF_TEXT_INDICES	INQUIRE LIST OF TEXT INDICES
INQ_LOCATOR_DEVICE_STATE	INQUIRE LOCATOR DEVICE STATE
INQ_MAX_LENGTH_OF_WS_STATE_TABLES	INQUIRE MAXIMUM LENGTH OF WORKSTATION STATE TABLES
INQ_MAX_NORMALIZATION_TRANSFORMATION_NUMBER	INQUIRE MAXIMUM NORMALIZATION TRANSFORMATION NUMBER
INQ_MORE_SIMULTANEOUS_EVENTS	INQUIRE MORE SIMULTANEOUS EVENTS
INQ_NAME_OF_OPEN_SEGMENT	INQUIRE NAME OF OPEN SEGMENT
INQ_NORMALIZATION_TRANSFORMATION	INQUIRE NORMALIZATION TRANSFORMATION
INQ_NUMBER_OF_AVAILABLE_LOGICAL_INPUT_DEVICES	INQUIRE NUMBER OF AVAILABLE LOGICAL INPUT DEVICES
INQ_NUMBER_OF_SEGMENT_PRIORITIES_SUPPORTED	INQUIRE NUMBER OF SEGMENT PRIORITIES SUPPORTED
INQ_OPERATING_STATE_VALUE	INQUIRE OPERATING STATE VALUE
INQ_PATTERN_FACILITIES	INQUIRE PATTERN FACILITIES
INQ_PATTERN_REPRESENTATION	INQUIRE PATTERN REPRESENTATION
INQ_PICK_DEVICE_STATE	INQUIRE PICK DEVICE STATE
INQ_PIXEL	INQUIRE PIXEL
INQ_PIXEL_ARRAY	INQUIRE PIXEL ARRAY
INQ_PIXEL_ARRAY_DIMENSION	INQUIRE PIXEL ARRAY DIMENSIONS
INQ_POLYLINE_FACILITIES	INQUIRE POLYLINE FACILITIES
INQ_POLYLINE_REPRESENTATION	INQUIRE POLYLINE REPRESENTATION
INQ_POLYMARKER_FACILITIES	INQUIRE POLYMARKER FACILITIES

Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name

<u>Ada Bound Name</u>	<u>GKS Function</u>
INQ_POLYMARKER_REPRESENTATION	INQUIRE POLYMARKER REPRESENTATION
INQ_PREDEFINED_COLOUR_REPRESENTATION	INQUIRE PREDEFINED COLOUR REPRESENTATION
INQ_PREDEFINED_FILL_AREA_REPRESENTATION	INQUIRE PREDEFINED FILL AREA REPRESENTATION
INQ_PREDEFINED_PATTERN_REPRESENTATION	INQUIRE PREDEFINED PATTERN REPRESENTATION
INQ_PREDEFINED_POLYLINE_REPRESENTATION	INQUIRE PREDEFINED POLYLINE REPRESENTATION
INQ_PREDEFINED_POLYMARKER_REPRESENTATION	INQUIRE PREDEFINED POLYMARKER REPRESENTATION
INQ_PREDEFINED_TEXT_REPRESENTATION	INQUIRE PREDEFINED TEXT REPRESENTATION
INQ_SEGMENT_ATTRIBUTES	INQUIRE SEGMENT ATTRIBUTES
INQ_SET_OF_ACTIVE_WS	INQUIRE SET OF ACTIVE WORKSTATIONS
INQ_SET_OF_ASSOCIATED_WS	INQUIRE SET OF ASSOCIATED WORKSTATIONS
INQ_SET_OF_OPEN_WS	INQUIRE SET OF OPEN WORKSTATIONS
INQ_SET_OF_SEGMENT_NAMES_IN_USE	INQUIRE SET OF SEGMENT NAMES IN USE
INQ_SET_OF_SEGMENT_NAMES_ON_WS	INQUIRE SET OF SEGMENT NAMES ON WORKSTATION
INQ_STRING_DEVICE_STATE	INQUIRE STRING DEVICE STATE
INQ_STROKE_DEVICE_STATE	INQUIRE STROKE DEVICE STATE
INQ_TEXT_EXTENT	INQUIRE TEXT EXTENT
INQ_TEXT_FACILITIES	INQUIRE TEXT FACILITIES
INQ_TEXT_REPRESENTATION	INQUIRE TEXT REPRESENTATION
INQ_VALUATOR_DEVICE_STATE	INQUIRE VALUATOR DEVICE STATE
INQ_WS_CATEGORY	INQUIRE WORKSTATION CATEGORY
INQ_WS_CLASSIFICATION	INQUIRE WORKSTATION CLASSIFICATION
INQ_WS_CONNECTION_AND_TYPE	INQUIRE WORKSTATION CONNECTION AND TYPE
INQ_WS_DEFERRAL_AND_UPDATE_STATES	INQUIRE WORKSTATION DEFERRAL AND UPDATE STATES
INQ_WS_MAX_NUMBERS	INQUIRE WORKSTATION MAXIMUM NUMBER
INQ_WS_STATE	INQUIRE WORKSTATION STATE
INQ_WS_TRANSFORMATION	INQUIRE WORKSTATION TRANSFORMATION
INSERT_SEGMENT	INSERT SEGMENT
INTERPRET_ITEM	INTERPRET ITEM
MESSAGE	MESSAGE
OPEN_GKS	OPEN GKS
OPEN_WS	OPEN WORKSTATION
POLYLINE	POLYLINE
POLYMARKER	POLYMARKER
READ_ITEM_FROM_GKSM	READ ITEM FROM GKSM
REDRAW_ALL_SEGMENTS_ON_WS	REDRAW ALL SEGMENTS ON WORKSTATION
RENAME_SEGMENT	RENAME SEGMENT

Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name

<u>Ada Bound Name</u>	<u>GKS Function</u>
REQUEST_CHOICE	REQUEST CHOICE
REQUEST_LOCATOR	REQUEST LOCATOR
REQUEST_PICK	REQUEST PICK
REQUEST_STRING	REQUEST STRING
REQUEST_STROKE	REQUEST STROKE
REQUEST_VALUATOR	REQUEST VALUATOR
SAMPLE_CHOICE	SAMPLE CHOICE
SAMPLE_LOCATOR	SAMPLE LOCATOR
SAMPLE_PICK	SAMPLE PICK
SAMPLE_STRING	SAMPLE STRING
SAMPLE_STROKE	SAMPLE STROKE
SAMPLE_VALUATOR	SAMPLE VALUATOR
SELECT_NORMALIZATION_TRANSFORMATION	SELECT NORMALIZATION TRANSFORMATION
SET_ASF	SET ASPECT SOURCE FLAGS
SET_CHAR_EXPANSION_FACTOR	SET CHARACTER EXPANSION FACTOR
SET_CHAR_HEIGHT	SET CHARACTER HEIGHT
SET_CHAR_SPACING	SET CHARACTER SPACING
SET_CHAR_UP_VECTOR	SET CHARACTER UP VECTOR
SET_CHOICE_MODE	SET CHOICE MODE
SET_CLIPPING_INDICATOR	SET CLIPPING INDICATOR
SET_COLOUR_REPRESENTATION	SET COLOUR REPRESENTATION
SET_DEFERRAL_STATE	SET DEFERRAL STATE
SET_DETECTABILITY	SET DETECTABILITY
SET_FILL_AREA_COLOUR_INDEX	SET FILL AREA COLOUR INDEX
SET_FILL_AREA_INDEX	SET FILL AREA INDEX
SET_FILL_AREA_INTERIOR_STYLE	SET FILL AREA INTERIOR STYLE
SET_FILL_AREA_REPRESENTATION	SET FILL AREA REPRESENTATION
SET_FILL_AREA_STYLE_INDEX	SET FILL AREA STYLE INDEX
SET_HIGHLIGHTING	SET HIGHLIGHTING
SET_LINETYPE	SET LINETYPE
SET_LINEWIDTH_SCALE_FACTOR	SET LINEWIDTH SCALE FACTOR
SET_LOCATOR_MODE	SET LOCATOR MODE
SET_MARKER_SIZE_SCALE_FACTOR	SET MARKER SIZE SCALE FACTOR
SET_MARKER_TYPE	SET MARKER TYPE
SET_PATTERN_REFERENCE_POINT	SET PATTERN REFERENCE POINT
SET_PATTERN_REPRESENTATION	SET PATTERN REPRESENTATION
SET_PATTERN_SIZE	SET PATTERN SIZE
SET_PICK_ID	SET PICK IDENTIFIER
SET_PICK_MODE	SET PICK MODE
SET_POLYLINE_COLOUR_INDEX	SET POLYLINE COLOUR INDEX
SET_POLYLINE_INDEX	SET POLYLINE INDEX
SET_POLYLINE_REPRESENTATION	SET POLYLINE REPRESENTATION
SET_POLYMARKER_COLOUR_INDEX	SET POLYMARKER COLOUR INDEX
SET_POLYMARKER_INDEX	SET POLYMARKER INDEX
SET_POLYMARKER_REPRESENTATION	SET POLYMARKER REPRESENTATION
SET_SEGMENT_PRIORITY	SET SEGMENT PRIORITY
SET_SEGMENT_TRANSFORMATION	SET SEGMENT TRANSFORMATION

Table 3 — GKS Function Names and Ada Names Ordered Alphabetically by Ada Bound Name

<u>Ada Bound Name</u>	<u>GKS Function</u>
SET_STRING_MODE	SET STRING MODE
SET_STROKE_MODE	SET STROKE MODE
SET_TEXT_ALIGNMENT	SET TEXT ALIGNMENT
SET_TEXT_COLOUR_INDEX	SET TEXT COLOUR INDEX
SET_TEXT_FONT_AND_PRECISION	SET TEXT FONT AND PRECISION
SET_TEXT_INDEX	SET TEXT INDEX
SET_TEXT_PATH	SET TEXT PATH
SET_TEXT_REPRESENTATION	SET TEXT REPRESENTATION
SET_VALUATOR_MODE	SET VALUATOR MODE
SET_VIEWPORT	SET VIEWPORT
SET_VIEWPORT_INPUT_PRIORITY	SET VIEWPORT INPUT PRIORITY
SET_VISIBILITY	SET VISIBILITY
SET_WINDOW	SET WINDOW
SET_WS_VIEWPORT	SET WORKSTATION VIEWPORT
SET_WS_WINDOW	SET WORKSTATION WINDOW
TEXT	TEXT
UPDATE_WS	UPDATE WORKSTATION
WRITE_ITEM_TO_GKSM	WRITE ITEM TO GKSM
Alphabetical GKS functions	
The functions are in the same order when listed alphabetically according to the GKS function names as they are when listed alphabetically according to the bound names. Therefore, Table 3, provided above, alphabetically lists the GKS functions.	

Table 4 — Alphabetical list of GKS functions by level

LEVEL 0A
ACTIVATE_WS
CELL_ARRAY
CLEAR_WS
CLOSE_GKS
CLOSE_WS
DEACTIVATE_WS
EMERGENCY_CLOSE_GKS
ERROR_HANDLING
ERROR_LOGGING
ESCAPE
FILL_AREA
GDP
GET_ITEM_TYPE_FROM_GKSM
INQ_CLIPPING
INQ_COLOUR_FACILITIES
INQ_COLOUR_REPRESENTATION

Table 4 — Alphabetical list of GKS functions by level

INQ_CURRENT_INDIVIDUAL_ATTRIBUTE_VALUES

The following functions are a one-to-many mapping of the GKS function “inquire current individual attribute values”:

INQ_CHAR_EXPANSION_FACTOR
 INQ_CHAR_SPACING
 INQ_FILL_AREA_COLOUR_INDEX
 INQ_FILL_AREA_INTERIOR_STYLE
 INQ_FILL_AREA_STYLE_INDEX
 INQ_LINETYPE
 INQ_LINEWIDTH_SCALE_FACTOR
 INQ_LIST_OF_ASF
 INQ_POLYLINE_COLOUR_INDEX
 INQ_POLYMARKER_COLOUR_INDEX
 INQ_POLYMARKER_SIZE_SCALE_FACTOR
 INQ_POLYMARKER_TYPE
 INQ_TEXT_COLOUR_INDEX
 INQ_TEXT_FONT_AND_PRECISION

INQ_CURRENT_NORMALIZATION_TRANSFORMATION_NUMBER

INQ_CURRENT_PRIMITIVE_ATTRIBUTE_VALUES

The following functions are a one-to-many mapping of the GKS function “inquire current primitive attribute values”:

INQ_POLYLINE_INDEX
 INQ_POLYMARKER_INDEX
 INQ_TEXT_INDEX
 INQ_CHAR_HEIGHT
 INQ_CHAR_UP_VECTOR
 INQ_CHAR_WIDTH
 INQ_CHAR_BASE_VECTOR
 INQ_TEXT_PATH
 INQ_TEXT_ALIGNMENT
 INQ_FILL_AREA_INDEX
 INQ_PATTERN_HEIGHT_VECTOR
 INQ_PATTERN_WIDTH_VECTOR
 INQ_PATTERN_REFERENCE_POINT

INQ_DISPLAY_SPACE_SIZE
 INQ_FILL_AREA_FACILITIES
 INQ_GDP
 INQ_LEVEL_OF_GKS
 INQ_LIST_OF_AVAILABLE_GDP
 INQ_LIST_OF_AVAILABLE_WS_TYPES
 INQ_LIST_OF_COLOUR_INDICES
 INQ_LIST_OF_NORMALIZATION_TRANSFORMATION_NUMBER
 INQ_MAX_LENGTH_WS_STATE_TABLES
 INQ_MAX_NORMALIZATION_TRANSFORMATION_NUMBER
 INQ_NORMALIZATION_TRANSFORMATION
 INQ_OPERATING_STATE_VALUE

Table 4 — Alphabetical list of GKS functions by level

INQ_PATTERN_FACILITIES
 INQ_PIXEL
 INQ_PIXEL_ARRAY
 INQ_PIXEL_ARRAY_DIMENSIONS
 INQ_POLYLINE_FACILITIES
 INQ_POLYMARKER_FACILITIES
 INQ_PREDEFINED_COLOUR_REPRESENTATION
 INQ_PREDEFINED_FILL_AREA_REPRESENTATION
 INQ_PREDEFINED_PATTERN_REPRESENTATION
 INQ_PREDEFINED_POLYLINE_REPRESENTATION
 INQ_PREDEFINED_POLYMARKER_REPRESENTATION
 INQ_PREDEFINED_TEXT_REPRESENTATION
 INQ_SET_OF_OPEN_WS
 INQ_TEXT_EXTENT
 INQ_TEXT_FACILITIES
 INQ_WS_CATEGORIES
 INQ_WS_CLASSIFICATION
 INQ_WS_CONNECTION_AND_TYPE
 INQ_WS_DEFERRAL_AND_UPDATE_STATES
 INQ_WS_STATE
 INQ_WS_TRANSFORMATION
 INTERPRET_ITEM
 OPEN_GKS
 OPEN_WS
 POLYLINE
 POLYMARKER
 READ_ITEM_FROM_GKSM
 SELECT_NORMALIZATION_TRANSFORMATION
 SET_ASF
 SET_CHAR_EXPANSION_FACTOR
 SET_CHAR_HEIGHT
 SET_CHAR_SPACING
 SET_CHAR_UP_VECTOR
 SET_CLIPPING_INDICATOR
 SET_COLOUR_REPRESENTATION
 SET_FILL_AREA_COLOUR_INDEX
 SET_FILL_AREA_INDEX
 SET_FILL_AREA_INTERIOR_STYLE
 SET_FILL_AREA_STYLE_INDEX
 SET_LINETYPE
 SET_LINEWIDTH_SCALE_FACTOR
 SET_MARKER_SIZE_SCALE_FACTOR
 SET_MARKER_TYPE
 SET_PATTERN_REFERENCE_POINT
 SET_PATTERN_SIZE
 SET_POLYLINE_COLOUR_INDEX
 SET_POLYMARKER_COLOUR_INDEX
 SET_POLYLINE_INDEX
 SET_POLYMARKER_INDEX

Table 4 — Alphabetical list of GKS functions by level

SET_TEXT_ALIGNMENT
 SET_TEXT_COLOUR_INDEX
 SET_TEXT_FONT_AND_PRECISION
 SET_TEXT_INDEX
 SET_TEXT_PATH
 SET_VIEWPORT
 SET_WINDOW
 SET_WS_VIEWPORT
 SET_WS_WINDOW

TEXT

UPDATE_WS

WRITE_ITEM_TO_GKSM

LEVEL 0b

INITIALISE_CHOICE
 INITIALISE_LOCATOR
 INITIALISE_STRING
 INITIALISE_STROKE
 INITIALISE_VALUATOR
 INQ_CHOICE_DEVICE_STATE
 INQ_DEFAULT_CHOICE_DEVICE_STATE
 INQ_DEFAULT_LOCATOR_DEVICE_DATA
 INQ_DEFAULT_STRING_DEVICE_DATA
 INQ_DEFAULT_STROKE_DEVICE_DATA
 INQ_DEFAULT_VALUATOR_DEVICE_DATA
 INQ_LOCATOR_DEVICE_STATE
 INQ_NUMBER_OF_AVAILABLE_LOGICAL_INPUT_DEVICES
 INQ_STRING_DEVICE_STATE
 INQ_STROKE_DEVICE_STATE
 INQ_VALUATOR_DEVICE_STATE
 REQUEST_CHOICE
 REQUEST_LOCATOR
 REQUEST_STRING
 REQUEST_STROKE
 REQUEST_VALUATOR
 SET_CHOICE_MODE
 SET_LOCATOR_MODE
 SET_STRING_MODE
 SET_STROKE_MODE
 SET_VALUATOR_MODE
 SET_VIEWPORT_INPUT_PRIORITY

LEVEL 0c

AWAIT_EVENT
 FLUSH_DEVICE_EVENTS
 GET_CHOICE
 GET_LOCATOR

Table 4 — Alphabetical list of GKS functions by level

GET_STRING
 GET_STROKE
 GET_VALUATOR
 INQ_INPUT_QUEUE_OVERFLOW
 INQ_MORE_SIMULTANEOUS_EVENTS
 SAMPLE_CHOICE
 SAMPLE_LOCATOR
 SAMPLE_STRING
 SAMPLE_STROKE
 SAMPLE_VALUATOR

LEVEL 1a

ACCUMULATE_TRANSFORMATION_MATRIX
 CLOSE_SEGMENT
 CREATE_SEGMENT
 DELETE_SEGMENT
 DELETE_SEGMENT_FROM_WS
 EVALUATE_TRANSFORMATION_MATRIX
 INQ_DEFAULT_DEFERRAL_STATE_VALUES
 INQ_DYNAMIC_MODIFICATION_OF_SEGMENT_ATTRIBUTES
 INQ_DYNAMIC_MODIFICATION_OF_WS_ATTRIBUTES
 INQ_FILL_AREA_REPRESENTATION
 INQ_LIST_OF_FILL_AREA_INDICES
 INQ_LIST_OF_PATTERN_INDICES
 INQ_LIST_OF_POLYLINE_INDICES
 INQ_LIST_OF_POLYMARKER_INDICES
 INQ_LIST_OF_TEXT_INDICES
 INQ_NAME_OF_OPEN_SEGMENT
 INQ_NUMBER_OF_SEGMENT_PRIORITIES_SUPPORTED
 INQ_PATTERN_REPRESENTATION
 INQ_POLYLINE_REPRESENTATION
 INQ_POLYMARKER_REPRESENTATION
 INQ_SEGMENT_ATTRIBUTES
 INQ_SET_OF_ACTIVE_WS
 INQ_SET_OF_ASSOCIATED_WS
 INQ_SET_OF_SEGMENT_NAMES_IN_USE
 INQ_SET_OF_SEGMENT_NAMES_ON_WS
 INQ_TEXT_REPRESENTATION
 INQ_WS_MAX_NUMBERS
 MESSAGE
 REDRAW_ALL_SEGMENTS_ON_WS
 RENAME_SEGMENT
 SET_DEFERRAL_STATE
 SET_FILL_AREA_REPRESENTATION
 SET_HIGHLIGHTING
 SET_PATTERN_REPRESENTATION
 SET_POLYLINE_REPRESENTATION
 SET_POLYMARKER_REPRESENTATION
 SET_SEGMENT_PRIORITY

Table 4 — Alphabetical list of GKS functions by level

	SET_SEGMENT_TRANSFORMATION
	SET_TEXT_REPRESENTATION
	SET_VISIBILITY
LEVEL 1b	
	INITIALISE_PICK
	INQ_CURRENT_PICK_ID_VALUE
	INQ_DEFAULT_PICK_DEVICE_DATA
	INQ_PICK_DEVICE_STATE
	REQUEST_PICK
	SET_DETECTABLILITY
	SET_PICK_ID
	SET_PICK_MODE
LEVEL 1c	
	GET_PICK
	SAMPLE_PICK
LEVEL 2a	
	ASSOCIATE_SEGMENT_WITH_WS
	COPY_SEGMENT_TO_WS
	INSERT_SEGMENT
LEVEL 2b	
	NONE
LEVEL 2c	
	NONE

4.2 Data Type Definitions

4.2.1 Abbreviations used in the data type definitions

ASF	aspect source flag
CHAR	character
DC	device coordinate
GDP	generalized drawing primitive
GKS	graphical kernel system
GKSM	graphical kernel system metafile
ID	identifier
MAX	maximum
NDC	normalized device coordinates
WC	world coordinate
WS	workstation

4.2.2 Alphabetical list of type definitions

This section contains an alphabetical listing of all of the data type definitions used to define the Ada binding to GKS. Each of these declarations specifies the level of GKS at which the data type declaration must be available in an implementation of GKS of that level or any level “above” the level in which the type declaration is first needed (same as for functions). Each element declared also includes a comment about the type and/or use of the type. Some of the declarations employ constant values in the definition of the type. All of these constant declarations are included in the GKS_TYPES package.

ASF	Level 0a
type ASF is (BUNDLED, INDIVIDUAL);	
-- This type defines an aspect source flag whose value indicates whether	
-- an aspect of a primitive should be set from a bundle table or from an	
-- individual attribute.	

ASF_LIST	Level 0a
type ASF_LIST is	
record	
TYPE_OF_LINE_ASF	: ASF;
WIDTH_ASF	: ASF;
LINE_COLOUR_ASF	: ASF;
TYPE_OF_MARKER_ASF	: ASF;
SIZE_ASF	: ASF;
MARKER_COLOUR_ASF	: ASF;
FONT_PRECISION_ASF	: ASF;
EXPANSION_ASF	: ASF;
SPACING_ASF	: ASF;
TEXT_COLOUR_ASF	: ASF;
INTERIOR_ASF	: ASF;
STYLE_ASF	: ASF;
FILL_AREA_COLOUR_ASF	: ASF;
end record;	
-- A record containing all of the aspect source flags, with components indicating the	
-- specific flag.	

ATTRIBUTES_USED

Level 0a

package ATTRIBUTES_USED is
 new GKS_LIST_UTILITIES (ATTRIBUTES_USED_TYPE);
 -- Provides for a list of the attributes used.

ATTRIBUTES_USED_TYPE

Level 0a

type ATTRIBUTES_USED_TYPE is (POLYLINE_ATTRIBUTES,
 POLYMARKER_ATTRIBUTES,
 TEXT_ATTRIBUTES,
 FILL_AREA_ATTRIBUTES);
 -- The types of attributes which may be used in generating output for a GDP and in
 -- generating prompt and echo information for certain prompt and echo types of
 -- certain classes of input devices.

CHAR_EXPANSION

Level 0a

type CHAR_EXPANSION is new SCALE_FACTOR range
 SCALE_FACTOR'SAFE_SMALL..SCALE_FACTOR'LAST';
 -- Defines a character expansion factor. Factors are unitless, and must be greater than zero.

CHAR_SPACING

Level 0a

type CHAR_SPACING is new SCALE_FACTOR;
 -- Defines a character spacing factor. The factors are unitless. A positive value indicates
 -- the amount of extra space between characters in a text string, and a negative value
 -- indicates the amount of overlap between character boxes in a text string.

CHOICE_DEVICE_NUMBER

Level 0b

type CHOICE_DEVICE_NUMBER is new DEVICE_NUMBER;
 -- Provides for choice device identifiers.

CHOICE_PROMPT	Level 0b
---------------	----------

type CHOICE_PROMPT is (OFF, ON);
 -- Indicates for a choice prompt and echo type whether a specified prompt is to
 -- be displayed or not.

CHOICE_PROMPTS	Level 0b
----------------	----------

package CHOICE_PROMPTS is
 new GKS_LIST_UTILITIES (CHOICE_PROMPT);
 -- Provides for lists of prompts.

CHOICE_PROMPT_ECHO_TYPE	Level 0b
-------------------------	----------

type CHOICE_PROMPT_ECHO_TYPE is new INTEGER;
 -- Defines the choice prompt and echo type.

CHOICE_PROMPT_ECHO_TYPES	Level 0b
--------------------------	----------

package CHOICE_PROMPT_ECHO_TYPES is
 new GKS_LIST_UTILITIES (CHOICE_PROMPT_ECHO_TYPE);
 -- Provides for lists of choice prompt and echo types.

CHOICE_PROMPT_STRING	Level 0b
----------------------	----------

type CHOICE_PROMPT_STRING (LENGTH: STRING_SMALL_NATURAL := 0) is
 record
 CONTENTS: STRING (1..LENGTH);
 end record;
 -- Provides for a variable length prompt. Objects of type should be declared
 -- unconstrained to allow for dynamic modification of the length.

CHOICE_PROMPT_STRING_ARRAY	Level 0b
----------------------------	----------

type CHOICE_PROMPT_STRING_ARRAY is array (POSITIVE range <>)
 of CHOICE_PROMPT_STRING;
 -- Provides for an array of prompt strings.

CHOICE_PROMPT_STRING_LIST	Level 0b
type CHOICE_PROMPT_STRING_LIST(LENGTH:CHOICE_SMALL_NATURAL:= 0) is record LIST : CHOICE_PROMPT_STRING_ARRAY (1..LENGTH); end record; -- Provides for lists of prompt strings.	
CHOICE_REQUEST_STATUS	Level 0b
type CHOICE_REQUEST_STATUS is (OK, NOCHOICE, NONE); -- Defines the status of a choice input operation for the request function.	
CHOICE_SMALL_NATURAL	Level 0b
subtype CHOICE_SMALL_NATURAL is NATURAL RANGE 0..CHOICE_SMALL_NATURAL_MAX; -- This is a subtype declaration which allows for unconstrained record -- objects for CHOICE_PROMPT_STRING_LIST type without causing the -- exception STORAGE_ERROR to be raised.	
CHOICE_STATUS	Level 0b
subtype CHOICE_STATUS is CHOICE_REQUEST_STATUS range OK..NOCHOICE; -- Indicates if a choice was made by the operator for the sample, get, and inquiry functions.	
CHOICE_VALUE	Level 0b
type CHOICE_VALUE is new POSITIVE; -- Defines the choice values available on an implementation.	
CLIPPING_INDICATOR	Level 0a
type CLIPPING_INDICATOR is (CLIP, NOCLIP); -- Indicates whether or not clipping is to be performed.	

COLOUR_AVAILABLE	Level 0a
------------------	----------

type COLOUR_AVAILABLE is (COLOUR, MONOCHROME);
 -- Indicates whether colour output is available on a workstation.

COLOUR_INDEX	Level 0a
--------------	----------

subtype COLOUR_INDEX is PIXEL_COLOUR_INDEX
 range 0..PIXEL_COLOUR_INDEX'LAST;
 -- Indices into colour tables are of this type.

COLOUR_INDICES	Level 0a
----------------	----------

package COLOUR_INDICES is new GKS_LIST_UTILITIES (COLOUR_INDEX);
 -- Provides for a set of colour indices which are available on a particular workstation.

COLOUR_MATRIX	Level 0a
---------------	----------

type COLOUR_MATRIX is array (POSITIVE range <>, POSITIVE range <>)
 of COLOUR_INDEX;
 -- Provides for matrices containing colour indices corresponding to a
 -- cell array or pattern array.

COLOUR_REPRESENTATION	Level 0a
-----------------------	----------

type COLOUR_REPRESENTATION is
 record
 RED : INTENSITY;
 GREEN : INTENSITY;
 BLUE : INTENSITY;
 end record;
 -- Defines the representation of a colour as a combination of intensities in
 -- an RGB colour system.

CONTROL_FLAG	Level 0a
--------------	----------

type CONTROL_FLAG is (CONDITIONALLY, ALWAYS);
 -- The control flag is used to indicate the conditions under which the display
 -- surface should be cleared.

DC	Level 0a
----	----------

package DC is new GKS_COORDINATE_SYSTEM (DC_TYPE);
 -- Defines the Device Coordinate System.

DC_TYPE	Level 0a
---------	----------

type DC_TYPE is digits PRECISION;
 -- The type of a coordinate in the Device Coordinate System.

DC_UNITS	Level 0a
----------	----------

type DC_UNITS is (METRES, OTHER);
 -- Device coordinate units for a particular workstation should be in metres unless
 -- the device is incapable of producing a precisely scaled image, and appropriate
 -- workstation dependent units otherwise.

DEFERRAL_MODE	Level 0a
---------------	----------

type DEFERRAL_MODE is (ASAP, BNIG, BNIL, ASTI);
 -- Defines the four GKS deferral modes.

DEVICE_NUMBER	Level 0b
---------------	----------

package DEVICE_NUMBER_TYPE is
 DEVICE_NUMBER_TYPE is new POSITIVE;
 end DEVICE_NUMBER_TYPE;
 -- Logical input devices are referenced as device numbers.

DISPLAY_CLASS	Level 0a
---------------	----------

type DISPLAY_CLASS is (VECTOR_DISPLAY,
 RASTER_DISPLAY,
 OTHER_DISPLAY);
 -- The classification of a workstation of category OUTPUT or OUTIN.

DISPLAY_SURFACE_EMPTY

Level 0a

type DISPLAY_SURFACE_EMPTY is (EMPTY, NOTEMPTY);
 -- Indicates whether the display surface is empty.

DYNAMIC_MODIFICATION

Level 1a

type DYNAMIC_MODIFICATION is (IRG, IMM);
 -- Indicates whether an update to the state list is performed immediately (IMM)
 -- or requires implicit regeneration (IRG).

ECHO_SWITCH

Level 0b

type ECHO_SWITCH is (ECHO, NOECHO);
 -- Indicates whether or not echoing of the prompt is performed.

ERROR_NUMBER

Level 0a

type ERROR_NUMBER is new INTEGER;
 -- Defines the type for error indicator values.

EVENT_DEVICE_NUMBER

Level 0c

type EVENT_DEVICE_NUMBER (CLASS: INPUT_CLASS := NONE) is
 record
 case CLASS is
 when NONE => null;
 when LOCATOR_INPUT => LOCATOR_EVENT_DEVICE
 : LOCATOR_DEVICE_NUMBER;
 when STROKE_INPUT => STROKE_EVENT_DEVICE
 : STROKE_DEVICE_NUMBER;
 when VALUATOR_INPUT => VALUATOR_EVENT_DEVICE
 : VALUATOR_DEVICE_NUMBER;
 when CHOICE_INPUT => CHOICE_EVENT_DEVICE
 : CHOICE_DEVICE_NUMBER;
 when PICK_INPUT => PICK_EVENT_DEVICE
 : PICK_DEVICE_NUMBER;
 when STRING_INPUT => STRING_EVENT_DEVICE
 : STRING_DEVICE_NUMBER;
 end case;
 end record;
 -- Provides for returning any class of device number from the event queue.

EVENT_OVERFLOW_DEVICE_NUMBER

Level 0c

```
type EVENT_OVERFLOW_DEVICE_NUMBER
  (CLASS: INPUT_QUEUE_CLASS := LOCATOR_INPUT) is
  record
    case CLASS is
      when LOCATOR_INPUT    => LOCATOR_EVENT_DEVICE
                             : LOCATOR_DEVICE_NUMBER;
      when STROKE_INPUT     => STROKE_EVENT_DEVICE
                             : STROKE_DEVICE_NUMBER;
      when VALUATOR_INPUT   => VALUATOR_EVENT_DEVICE
                             : VALUATOR_DEVICE_NUMBER;
      when CHOICE_INPUT     => CHOICE_EVENT_DEVICE
                             : CHOICE_DEVICE_NUMBER;
      when PICK_INPUT       => PICK_EVENT_DEVICE
                             : PICK_DEVICE_NUMBER;
      when STRING_INPUT     => STRING_EVENT_DEVICE
                             : STRING_DEVICE_NUMBER;
    end case;
  end record;
```

FILL_AREA_INDEX

Level 0a

```
type FILL_AREA_INDEX is new POSITIVE;
-- Defines fill area bundle table indices.
```

FILL_AREA_INDICES

Level 0a

```
package FILL_AREA_INDICES is
  new GKS_LIST_UTILITIES (FILL_AREA_INDEX);
-- Provides for lists of fill area bundle table indices.
```

GDP_ID

Level 0a

```
type GDP_ID is new INTEGER;
-- Selects among the kinds of Generalized Drawing Primitives.
```

GDP_IDS

Level 0a

```
package GDP_IDS is new GKS_LIST_UTILITIES (GDP_ID);
-- Provides for lists of Generalized Drawing Primitive IDs.
```

GKS_Level

Level 0a

type GKS_Level is (L0a, L0b, L0c, L1a, L1b, L1c, L2a, L2b, L2c);
 -- The valid Levels of GKS.

GKSM_ITEM_TYPE

Level 0a

type GKSM_ITEM_TYPE is new NATURAL;
 -- The type of an item contained in a GKSM metafile.

HATCH_STYLE

Level 0a

subtype HATCH_STYLE is STYLE_INDEX;
 -- Defines the fill area hatch styles type.

HATCH_STYLES

Level 0a

package HATCH_STYLES is new GKS_LIST_UTILITIES (HATCH_STYLE);
 -- Provides for lists of hatch styles.

HORIZONTAL_ALIGNMENT

Level 0a

type HORIZONTAL_ALIGNMENT is (NORMAL, LEFT, CENTRE, RIGHT);
 -- The alignment of the text extent parallelogram with respect to the horizontal
 -- positioning of the text.

IMPLEMENTATION_DEFINED_ERROR

Level 0a

subtype IMPLEMENTATION_DEFINED_ERROR is ERROR_NUMBER
 range ERROR_NUMBER'FIRST .. - 1;
 -- Defines the range of ERROR_NUMBERS to indicate that an implementation
 -- defined error has occurred.

INDIVIDUAL_ATTRIBUTE_VALUES

Level 0a

type INDIVIDUAL_ATTRIBUTE_VALUES is

record

TYPE_OF_LINE	: LINETYPE;
WIDTH	: LINEWIDTH;
LINE_COLOUR	: COLOUR_INDEX;
TYPE_OF_MARKER	: MARKER_TYPE;
SIZE	: MARKER_SIZE;
MARKER_COLOUR	: COLOUR_INDEX;
FONT_PRECISION	: TEXT_FONT_PRECISION;
EXPANSION	: CHAR_EXPANSION;
SPACING	: CHAR_SPACING;
TEXT_COLOUR	: COLOUR_INDEX;
INTERIOR	: INTERIOR_STYLE;
STYLE	: STYLE_INDEX;
FILL_AREA_COLOUR	: COLOUR_INDEX;
ASF	: ASF_LIST;

end record;

-- A record containing all of the current individual attributes for the procedure

-- INQ_CURRENT_INDIVIDUAL_ATTRIBUTE_VALUES.

INPUT_CLASS

Level 0b

type INPUT_CLASS is (NONE,
LOCATOR_INPUT,
STROKE_INPUT,
VALUATOR_INPUT,
CHOICE_INPUT,
PICK_INPUT
STRING_INPUT);

-- Defines the input device classifications for workstations of category INPUT or OUTIN.

INPUT_QUEUE_CLASS

Level 0c

subtype INPUT_QUEUE_CLASS is INPUT_CLASS range
LOCATOR_INPUT .. STRING_INPUT;

-- Defines the input device classifications for situations in which the

-- NONE classification is impossible.

INPUT_STATUS

Level 0b

type INPUT_STATUS is (OK, NONE);

-- Defines the status of a locator, stroke, valuator, or string operation.

INPUT_STRING

Level 0b

type INPUT_STRING (LENGTH: STRING_SMALL_NATURAL :=0) is
 record
 CONTENTS: STRING (1..LENGTH);
 end record;
 -- Provides a variable length string. Objects of this type should be declared
 -- unconstrained to allow for dynamic modification of the length.

INTENSITY

Level 0a

type INTENSITY is digits PRECISION range 0.0..1.0;
 -- Defines the range of possible intensities of a colour.

INTERIOR_STYLE

Level 0a

type INTERIOR_STYLE is (HOLLOW, SOLID, PATTERN, HATCH);
 -- Defines the fill area interior styles.

INTERIOR_STYLES

Level 0a

package INTERIOR_STYLES is
 new GKS_LIST_UTILITIES (INTERIOR_STYLE);
 -- Provides for lists of interior styles.

INVALID_VALUES_INDICATOR

Level 0a

type INVALID_VALUES_INDICATOR is (ABSENT, PRESENT);
 -- Indicates whether the value – 1 (i.e. “invalid”) is absent from or present
 -- in the PIXEL_ARRAY parameter returned by INQ_PIXEL_ARRAY.

LANGUAGE_BINDING_ERROR

Level 0a

subtype LANGUAGE_BINDING_ERROR is ERROR_NUMBER range 2500..2999;
 -- Defines the range of ERROR_NUMBERS to indicate a language binding error has occurred.

LINETYPE

Level 0a

type LINETYPE is new INTEGER;
 -- Defines the types of line styles provided by GKS.

LINETYPES	Level 0a
-----------	----------

package LINETYPES is new GKS_LIST_UTILITIES (LINETYPE);
 -- Provides for lists of line types.

LINEWIDTH	Level 0a
-----------	----------

type LINEWIDTH is new SCALE_FACTOR range 0.0..SCALE_FACTOR'LAST;
 -- The width of a line is indicated by a scale factor.

LOCATOR_DEVICE_NUMBER	Level 0b
-----------------------	----------

type LOCATOR_DEVICE_NUMBER is new DEVICE_NUMBER;
 -- Provides for locator device identifiers.

LOCATOR_PROMPT_ECHO_TYPE	Level 0b
--------------------------	----------

type LOCATOR_PROMPT_ECHO_TYPE is new INTEGER;
 -- Defines the locator prompt and echo types supported by the implementation.

LOCATOR_PROMPT_ECHO_TYPES	Level 0b
---------------------------	----------

package LOCATOR_PROMPT_ECHO_TYPES is
 new GKS_LIST_UTILITIES (LOCATOR_PROMPT_ECHO_TYPE);
 -- Provides for lists of locator prompt and echo types.

MARKER_SIZE	Level 0a
-------------	----------

type MARKER_SIZE is new SCALE_FACTOR range 0.0..SCALE_FACTOR'LAST;
 -- The size of a marker is indicated by a scale factor.

MARKER_TYPE	Level 0a
-------------	----------

type MARKER_TYPE is new INTEGER;
 -- Defines the type for markers provided by GKS.

MARKER_TYPES	Level 0a
--------------	----------

package MARKER_TYPES is new GKS_LIST_UTILITIES (MARKER_TYPE);
 -- Provides for lists of marker types.

MORE_EVENTS	Level 0c
-------------	----------

type MORE_EVENTS is (NOMORE, MORE);
 -- Indicates whether more events are contained in the input event queue.

NDC	Level 0a
-----	----------

package NDC is new GKS_COORDINATE_SYSTEM (NDC_TYPE);
 -- Defines the Normalized Device Coordinate System.

NDC_TYPE	Level 0a
----------	----------

type NDC_TYPE is digits PRECISION;
 -- Defines the type of a coordinate in the Normalized Device Coordinate System.

NEW_FRAME_NECESSARY	Level 0a
---------------------	----------

type NEW_FRAME_NECESSARY is (NO, YES);
 -- Indicates whether a new frame action is necessary at update.

OPERATING_MODE	Level 0b
----------------	----------

type OPERATING_MODE is (REQUEST_MODE, SAMPLE_MODE, EVENT_MODE);
 -- Defines the operating modes of an input device.

OPERATING_STATE	Level 0a
-----------------	----------

type OPERATING_STATE is (GKCL, GKOP, WSOP, WSAC, SGOP);
 -- Defines the five GKS operating states.

PATTERN_INDEX	Level 0a
subtype PATTERN_INDEX is STYLE_INDEX range 1..STYLE_INDEX'LAST; -- Defines the range of pattern table indices.	
PATTERN_INDICES	Level 0a
package PATTERN_INDICES is new GKS_LIST_UTILITIES (PATTERN_INDEX); -- Provides for lists of pattern table indices.	
PICK_DEVICE_NUMBER	Level 1b
type PICK_DEVICE_NUMBER is new DEVICE_NUMBER; -- Provides for pick devices.	
PICK_ID	Level 1b
type PICK_ID is new POSITIVE; -- Defines the range of pick identifiers available on an implementation.	
PICK_IDS	Level 1b
package PICK_IDS is new GKS_LIST_UTILITIES (PICK_ID); -- Provides for lists of pick identifiers.	
PICK_PROMPT_ECHO_TYPE	Level 0b
type PICK_PROMPT_ECHO_TYPE is new INTEGER; -- Defines the pick prompt and echo type.	
PICK_PROMPT_ECHO_TYPES	Level 0b
package PICK_PROMPT_ECHO_TYPES is new GKS_LIST_UTILITIES (PICK_PROMPT_ECHO_TYPE); -- Provides for lists of pick prompt and echo types.	

PICK_REQUEST_STATUS	Level 1b
---------------------	----------

type PICK_REQUEST_STATUS is (OK, NOPICK, NONE);
 -- Defines the status of a pick input operation for the request function.

PICK_STATUS	Level 1b
-------------	----------

subtype PICK_STATUS is PICK_REQUEST_STATUS range OK..NOPICK;
 -- Defines the status of a pick input operation for the sample, get, and inquiry functions.

PIXEL_COLOUR_INDEX	Level 0a
--------------------	----------

type PIXEL_COLOUR_INDEX is new INTEGER range – 1..INTEGER'LAST;
 -- A type for the pixel colour where the value – 1 represents an invalid colour index.

PIXEL_COLOUR_MATRIX	Level 0a
---------------------	----------

type PIXEL_COLOUR_MATRIX is array (POSITIVE range <>, POSITIVE range <>) of PIXEL_COLOUR_INDEX;
 -- Provides for matrices of pixel colours.

POLYLINE_INDEX	Level 0a
----------------	----------

type POLYLINE_INDEX is new POSITIVE;
 -- Defines the range of polyline indices.

POLYLINE_INDICES	Level 0a
------------------	----------

package POLYLINE_INDICES is
 new GKS_LIST_UTILITIES (POLYLINE_INDEX);
 -- Provides for lists of polyline indices.

POLYMARKER_INDEX

Level 0a

type POLYMARKER_INDEX is new POSITIVE;
-- Defines the range of polymarker bundle table indices.

POLYMARKER_INDICES

Level 0a

package POLYMARKER_INDICES is
new GKS_LIST_UTILITIES (POLYMARKER_INDEX);
-- Provides for lists of polymarker indices.

POSITIVE_TRANSFORMATION_NUMBER

Level 0a

subtype POSITIVE_TRANSFORMATION_NUMBER is TRANSFORMATION_NUMBER
range 1 .. TRANSFORMATION_NUMBER'LAST
-- A normalization transformation number corresponding to a settable transformation

PRIMITIVE_ATTRIBUTE_VALUES

Level 0a

type PRIMITIVE_ATTRIBUTE_VALUES is
record

INDEX_POLYLINE	: POLYLINE_INDEX;
INDEX_POLYMARKER	: POLYMARKER_INDEX;
INDEX_TEXT	: TEXT_INDEX;
CHAR_HEIGHT	: WC.MAGNITUDE;
CHAR_UP	: WC.VECTOR;
CHAR_WIDTH	: WC.MAGNITUDE;
CHAR_BASE	: WC.VECTOR;
PATH	: TEXT_PATH;
ALIGNMENT	: TEXT_ALIGNMENT;
INDEX_FILL_AREA	: FILL_AREA_INDEX;
PATTERN_WIDTH_VECTOR	: WC.VECTOR;
PATTERN_HEIGHT_VECTOR	: WC.VECTOR;
PATTERN_REFERENCE_POINT	: WC.POINT;

end record;

-- A record containing all of the current primitive attributes for the procedure
-- INQ_CURRENT_PRIMITIVE_ATTRIBUTE_VALUES.

RADIANS

Level 1a

type RADIANS is digits PRECISION;
-- Values used in performing segment transformations (rotation angle). Positive indicates
-- an anticlockwise direction.

RANGE_OF_EXPANSIONS

Level 0a

type RANGE_OF_EXPANSIONS is
 record
 MIN : CHAR_EXPANSION;
 MAX : CHAR_EXPANSION;
 end record;
-- Provides a range of character expansion factors.

RASTER_UNITS

Level 0a

type RASTER_UNITS is new POSITIVE;
-- Defines the range of raster units.

RASTER_UNIT_SIZE

Level 0a

type RASTER_UNIT_SIZE is
 record
 X : RASTER_UNITS;
 Y : RASTER_UNITS;
 end record;
-- Defines the size of a display screen in raster units on a raster device.

REGENERATION_MODE

Level 0a

type REGENERATION_MODE is (SUPPRESSED, ALLOWED);
-- Indicates whether implicit regeneration of the display is suppressed or allowed.

RELATIVE_PRIORITY	Level 0a
type RELATIVE_PRIORITY is (HIGHER, LOWER); -- Indicates the relative priority between two normalization transformations.	
RETURN_VALUE_TYPE	Level 0a
type RETURN_VALUE_TYPE is (SET, REALIZED); -- Indicates whether the returned values should be as they were set by the program or as they -- were actually realized on the device.	
SCALE_FACTOR	Level 0a
package SCALE_FACTOR_TYPE is type SCALE_FACTOR is digits PRECISION; end SCALE_FACTOR_TYPE; -- The type used for unitless scaling factors.	
SEGMENT_DETECTABILITY	Level 1a
type SEGMENT_DETECTABILITY is (UNDETECTABLE, DETECTABLE); -- Indicates whether a segment is detectable or not.	
SEGMENT_HIGHLIGHTING	Level 1a
type SEGMENT_HIGHLIGHTING is (NORMAL, HIGHLIGHTED); -- Indicates whether a segment is highlighted or not.	
SEGMENT_NAME	Level 1a
type SEGMENT_NAME is new POSITIVE; -- Defines the range of segment names.	

SEGMENT_NAMES	Level 1a
package SEGMENT_NAMES is new GKS_LIST_UTILITIES (SEGMENT_NAME); -- Provides for lists of segment names.	
SEGMENT_PRIORITY	Level 1a
type SEGMENT_PRIORITY is digits PRECISION range 0.0..1.0; -- Defines the priority of a segment.	
SEGMENT_VISIBILITY	Level 1a
type SEGMENT_VISIBILITY is (VISIBLE, INVISIBLE); -- Indicates whether a segment is visible or not.	
SMALL_NATURAL	Level 0a
subtype SMALL_NATURAL is NATURAL range 0..SMALL_NATURAL_MAX; -- This is a subtype declaration which allows for unconstrained record objects for various record -- types without causing the exception STORAGE_ERROR to be raised.	
STRING_DEVICE_NUMBER	Level 0b
type STRING_DEVICE_NUMBER is new DEVICE_NUMBER; -- Provides for string device number.	
STRING_PROMPT_ECHO_TYPE	Level 0b
type STRING_PROMPT_ECHO_TYPE is new INTEGER; -- Defines the string prompt and echo types.	

STRING_PROMPT_ECHO_TYPES

Level 0b

package STRING_PROMPT_ECHO_TYPES is
 new GKS_LIST_UTILITIES (STRING_PROMPT_ECHO_TYPE);
-- Provides for lists of string prompt and echo types.

STRING_SMALL_NATURAL

Level 0a

subtype STRING_SMALL_NATURAL is NATURAL
 range 0..STRING_SMALL_NATURAL_MAX;
-- This is a subtype declaration which allows for unconstrained record objects for various string
-- record types without causing the exception STORAGE_ERROR to be raised.

STROKE_DEVICE_NUMBER

Level 0b

type STROKE_DEVICE_NUMBER is new DEVICE_NUMBER;
-- Provides for stroke device numbers.

STROKE_PROMPT_ECHO_TYPE

Level 0b

type STROKE_PROMPT_ECHO_TYPE is new INTEGER;
-- Defines the stroke prompt and echo types.

STROKE_PROMPT_ECHO_TYPES

Level 0b

package STROKE_PROMPT_ECHO_TYPES is
 new GKS_LIST_UTILITIES (STROKE_PROMPT_ECHO_TYPE);
-- Provides for lists of stroke prompt and echo types.

STYLE_INDEX

Level 0a

type STYLE_INDEX is new INTEGER;
-- A style index is either a HATCH_STYLE or a PATTERN_STYLE.

TEXT_ALIGNMENT Level 0a

```
type TEXT_ALIGNMENT is
  record
    HORIZONTAL      : HORIZONTAL_ALIGNMENT;
    VERTICAL         : VERTICAL_ALIGNMENT;
  end record;
-- The type of the attribute controlling the positioning of the text extent parallelogram
-- in relation to the text position, having horizontal and vertical components as
-- defined above.
```

TEXT_EXTENT_PARALLELOGRAM Level 0a

```
type TEXT_EXTENT_PARALLELOGRAM is
  record
    LOWER_LEFT      : WC.POINT;
    LOWER_RIGHT     : WC.POINT;
    UPPER_RIGHT     : WC.POINT;
    UPPER_LEFT      : WC.POINT;
  end record;
-- Defines the corner points of the text extent parallelogram with respect to the
-- vertical positioning of the text.
```

TEXT_FONT Level 0a

```
type TEXT_FONT is new INTEGER;
-- Defines the types of fonts provided by the implementation.
```

TEXT_FONT_PRECISION Level 0a

```
type TEXT_FONT_PRECISION is
  record
    FONT            : TEXT_FONT;
    PRECISION       : TEXT_PRECISION;
  end record;
-- This type defines a record describing the text font and precision aspect.
```

TEXT_FONT_PRECISIONS	Level 0a
----------------------	----------

package TEXT_FONT_PRECISIONS is
new GKS_LIST_UTILITIES (TEXT_FONT_PRECISION);
-- Provides for lists of text font and precision pairs.

TEXT_INDEX	Level 0a
------------	----------

type TEXT_INDEX is new POSITIVE;
-- Defines the range of text bundle table indices.

TEXT_INDICES	Level 0a
--------------	----------

package TEXT_INDICES is new GKS_LIST_UTILITIES (TEXT_INDEX);
-- Provides for lists of text indices.

TEXT_PATH	Level 0a
-----------	----------

type TEXT_PATH is (RIGHT, LEFT, UP, DOWN);
-- The direction taken by a text string.

TEXT_PRECISION	Level 0a
----------------	----------

type TEXT_PRECISION is (STRING_PRECISION,
CHAR_PRECISION,
STROKE_PRECISION);
-- The precision with which text appears.

TRANSFORMATION_FACTOR	Level 1a
-----------------------	----------

type TRANSFORMATION_FACTOR is
record
X: NDC_TYPE;
Y: NDC_TYPE;
end record;
-- Scale factors used in building transformation matrices for performing segment
-- transformations.
