INTERNATIONAL STANDARD

ISO/IEC 10179

First edition
1996-04-01

AMENDMENT 1
2003-10-01

Information technology — Processing languages — Document Style Semantics and Specification Language (DSSSL)

AMENDMENT 1: Extensions to DSSSL

Technologies de l'information — Langages de traitement — Sémantique de présentation de documents et langage de spécifications (DSSSL)

AMENDEMENT 1: Extensions de DSSSL

Reference number
ISO/IEC 10179:1996/Amd.1:2003(E)

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

Published in Switzerland

Contents

Page

Foreword

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

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

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

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

Amendment 1 to ISO/IEC 10179:1996 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 34, *Document description and processing languages*.

Introduction

This Amendment specifies additional flow object classes and additional formatting areas, and clarifies some existing definitions in ISO/IEC 10179:1996, as follows.

a) Additional flow object classes are defined to specify on flow object classes for online display. Additional formatting areas are defined to clarify formatting areas for inline-display area and inter-line attachment area.

b) Annex B (informative) summarizes and clarifies the existing types and symbols. This annex will be a key to considering future extensions to DSSSL.

c) Annex C (informative) specifies formal public identifiers and their associated characteristics for various line breaking and line composition methods.

d) Annex D (informative) summarizes and clarifies the grove plan and SGML property set.

e) Annex E (informative) summarizes and clarifies the flow object classes and characteristics.

f) Annex F (informative) summarizes and clarifies the values of characteristics.

Information technology — Processing languages — Document Style Semantics and Specification Language (DSSSL)

AMENDMENT 1: Extensions to DSSSL

Abstract Application Program Interface to DSSSL Flow Object Tree

Add the following subclause after subclause 12.2.

12.2.1 API to Flow Object Tree

A DSSSL processor generates a flow object tree. The flow object tree contains information about the results of applying formatting specifications. DSSSL style editors operate on the flow object tree through an abstract application program interface. This API reports the following information:

— character, glyph and glyph-annotations

— line composition

— paragraph composition

— column-set composition

— page composition

— document composition

— documents' volume composition

The abstract API will include dynamic information relating to the document under construction. DSSSL will define a core interface for page composition. The API architecture is system independent.

Formatting Areas

Add the following subclauses after subclause 12.3.4.

12.3.5 Inline-display areas

DSSSL has the following conceptualized formatting areas:

— Display area

— Inline area

— Attachment area

Display areas may include other display areas as well as inline areas. Inline areas may also include conceptualized display areas known as inline-display areas. DSSSL defines an inline-display area as a "display area included within an inline area".

— display-area := (display-area* | inline-area*)

— inline-area := (inline-display-area* | contents*)

— inline-display-area := (inline-display-area* | contents*)

For examples, DSSSL has the following formatting areas for composition:

— (Main) Display area

  — (Sub) Display area

    — (Subsub) Display area ...

    — Inline area ...

      — Inline-display area ...

  — Inline area ...

    — Inline-display area ...

12.3.6 Inter-line attachment areas

The concept of an attachment creates a link between one of display area or inline area and an attachment area. The attachment area for a display area shall be outside the display area. The attachment area for an inline area shall be within the inter-line area between the current line and the immediately following line, or the border adjacent to the next display area. This amendment, therefore, extends the set of DSSSL formatting areas to the following set:

— Display area

— Inline area

— Inline-display area

— Inter-line area

— Attachment area (for display area)

— Inter-line attachment area

Flow Object Classes

Add the following subclauses after subclause 12.6.28.4.

12.6.28.5 Display-window Flow Object Class

This clause defines an extended-online feature. The display-window flow object class specifies an abstract size for the display frame of an online display. The display-window flow object class may get the top position of current scroll flow object classes as its root class. The display-window flow object class may be a recursive flow object class. It has a single principal port, which accepts any displayed flow objects. This flow object has the following characteristics:

— frame-type: one of the symbols window, dialogue, note, caution, alarm or warning. It specifies the type of window to be displayed. The default value is window.

— frame-size: one of the symbols maximum-size, optimum-size or minimum-size. It specifies the relative size of window to be used for an online display. The default value is optimum-size. This actual values used to represent this characteristic will depend on frame-type characteristics and display devices.

12.6.28.6 Pop-up Flow Object Class

This clause defines an extended-online feature. The pop-up flow object class provides information relating to a pop-up area, or the edge area of the current window frame. It has a single principal port, which accepts any displayed flow objects. This flow object has the following characteristics:

— information-type: one of the symbols anything, warning, error, additional-information, note, origin-of-link, voice-annotation or semantic-annotation. It specifies the type of information to be placed into the pop-up window or the edge area of the current window on online display. The default value is anything.

NOTE This flow object can be used to display any information, including position data relating to the grove structure.

12.6.28.7 Dialogue Flow Object Class

This clause defines an extended-online feature. The dialogue flow object class is used to specify interactive dialogues for online display. The dialogue flow object class shall be placed at the front top of the screen display within a display-window flow object class. Dialogue flow object classes may be treated as an interactive flow object class. It has a single principal port, which accepts any displayed flow objects. This dialogue flow object has the following characteristics:

— dialogue-type: one of the symbols request, acknowledgement, select-objects, select-from-list or interaction. The initial value is acknowledgement.

— dialogue-return-type: one of the symbols yes-no-cancel, yes-no, OK-NO, OK, tokens or phrase. The initial value is OK.

12.6.29 Sound-Voice and Animation Flow Object Classes

This clause defines the sound-voice-and-animation feature. The sound-voice and animation flow object classes is used for sounds, voices and animations stored within an external entity. Flow objects of these classes may be inlined or displayed as an online display. This flow object is atomic.

12.6.29.1 Sound-voice Flow Object Class

This clause defines the sound-voice feature. The sound-voice flow object class is used to specify sound and voice data to be used in conjunction with an online display. The sound-voice flow object class may be an atomic object on the other flow object classes, and it may be recursive on itself, depending on the sound-voice system being used. It has a single principal port, which accepts any flow objects. This flow object has the following characteristics:

— output?: boolean specifying whether the flow object shall be output or not. This characteristics is not inherited. The default value is #t.

Other specifications are system independent.

12.6.29.2 Animation Flow Object Class

This clause defines the animation feature. The animation flow object class is used to specify animation on an online display. Animation flow object class may be an atomic object on other flow object classes. It has a single principal port, which accepts any flow objects. This flow object has the following characteristics:

— output?:boolean specifying whether the flow object shall be output rather than displayed and inlined. This characteristics is not inherited. The default value is #t.

Other specifications are system independent.

Annexes

Add the following annexes after Annex A:

— *Annex B (informative) Types and Symbols*

— *Annex C (informative) Formal public identifiers and their associated characteristics*

— *Annex D (informative) Grove plan and SGML property set*

— *Annex E (informative) Flow object classes and characteristics*

— *Annex F (informative) Values of Characteristics*

Annex B
(informative)

Types and Symbols

This annex clarifies the existing data types of specification for DSSSL.

B.1 Expression Language

DSSSL expressions have the following data types:

boolean: logic-type

This type specifies true or false. The value can be defined whether #t or #f. This type is used in conditional expressions. #f means false, #t means true, other objects except #t and #f mean true. (See 8.2.2 True and False, and 8.5.1 Booleans.)

e.g., ((if equal wordflag #***t***) *x y*)

symbol: symbol-type

The symbol identifies strings for optional semantics. The value is used for specification of DSSSL scripts as parameters. In an expression, the symbol requires a single quotation (') before the string of symbol. (See 8.5.4 Symbols.)

e.g., (font-weight: '*bold*)

keyword: keyword-type

The keyword specifies names of keyword strings. This type is used for specification of a DSSSL script as constant and invariable parameters. In an expression, the keyword requires a single colon (:) after the strings of keywords and some keyword arguments. (See 8.5.5 Keywords.)

e.g., (font-weight: '***bold***)

char: character-type

The char specifies characters and names of characters. In an expression, the char requires a single sharp (#) and backslash (\) before the character or the name of character. (See 8.5.8 Characters.)

e.g.,

#**A**: character 'A'

#**space**: spacing 'empty character'

#**(**: open circle parentheses '('

pair: pair-type

The pair is constructed with two arguments. A list is constructed from a head part and other part of a pair; the head part is called **car**, other part is called **cdr**.

A pair is converted from a list. When a list is converted from a pair, DSSSL processor should generate an empty list as a tail of the list. The empty list is known as ().

e.g., (glyph-subset-table (list (cons *glyph1 glyph2*)))

DSSSL expressions have the following numerical types:

quantity: quantity-type

number: number-type

real: real-number-type

integer: integer-number-type

A quantity-type is represented as the product of a number and the base unit raised to the power of an integer-type. A number-type is the quantity-type without dimension. A real-number-type is a subtype of the number-type. An integer-type is a subtype of the real-number-type and integer-number-type. The quantity-type has both concepts of exactness and inexactness. An inexactness of the quantity-type allows a large size of numerical object to be printed or displayed.

e.g., (even? *tapesize*)

string: string-type

A string-type is similar to a string-type of other programming languages. The string-type is enclosed with double quotation (").

e.g., (string=? headstr "*Contents*")

procedure: procedure-type

A procedure-type is a name of procedure treated as an object in DSSSL.

e.g., (apply (***format-number*** *stringsizes*))

language: language-type

e.g., (with-language ***french*** (spellcheck-french))

B.2 SDQL

node-list: node-list-type

A node-list is the most basic type of SDQL that is used to specify node lists of a grove. A single node should be specified by node-list with a single node list. In DSSSL transformations, this data type with flag specifies and discriminates the result node-list. (See 10 Standard Document Query Language and 10.1.2 Node Lists.)

e.g., (node-list-first ***firstpage***)

named-node-list: node-list-with-name-type

A named-node-list specifies node lists with name in a grove tree. (See 10 Standard Document Query Language and 10.1.3 Named Node Lists.)

e.g., (named-node *column1* ***page-top***)

B.3 Transformation Language

subgrove-spec: sub-grove-specification-type

A subgrove-spec specifies a sub-grove. All the sub-grove can be specified by specifying the root of the sub-grove.

e.g., (subgrove-spec node | subgrove | class | add | null | remove | children | sublabel sort-children)

create-spec: create-specification-type

e.g.,

(**create-root** *obj sg*)

(**create-sub** *snl sg property label unique*)

(**create-proc** *snl sg label result-path optional unique*)

(**create-follow** *snl sg label result-path optional unique*)

result-node-list: result-node-list-type

e.g., (**select-by-relation** *rnl i proc*)

transform-grove-spec: transform-grove-specification-type

Transform-grove-spec specifies information of a grove transformation.

e.g., (**transform-grove** *snl obj ...*) (**select-grove** *nl obj*)

transliteration-map: transliteration-map-type

e.g., (**define-transliteration-map** *variable transliteration-entry*)

B.4 Style Language

sosofo: sosofo-type

A flow object class of DSSSL generates a specification of sequence of flow object(sosofo) as the result of execution.

e.g., (element p (**make** *paragraph*))

style: style type

e.g., (make paragraph style: ***emphasizing-style***)

generated-object: generated object type

e.g., (asis-indirect-sosofo (***column-number***))

length: length type

e.g., page-height: ***15in***

decoration-area: decoration area type

e.g.,

(make external-graphic entity-system-id: "sample.gif"
(decoration-area ***"graphics sample"***
placement-point-x: ***50***
placement-point-y: ***250***))

display-space: display space type

e.g., space-before: (display-space ***15pt max: 45pt priority: 1***)

inline-space: inline space type

e.g., escapement-space-before: (inline-space ***15pt max:45pt***)

glyph-id: glyph identifier type

e.g., (glyph-subst *gst-eng* ***glyph-aacute***)

glyph-subst-table: glyph substitution table type

e.g., (make character glyph-id: *eacute* glyph-subst-table: ***gst-eng***)

address: address type

e.g., (make *link* destination: ***chapter-2***)

color-space: color space type

(color-space string arg ...)

e.g., (color-space ***"ISO/IEC 10179:1996//Color-Space Family::Device RGB"***)

color: color type

(color color-space arg ...)

e.g.,

(define *rgb-color-space*
(color-space
"ISO/IEC 10179:1996//Color-Space Family::Device RGB"))
(define *midnight-blue* (color *rgb-color-space* 0.0 0.0 0.5))
...
(make paragraph color: ****midnight-blue****)

page-model: page-model type

(define-page-model ***page-model-name*** [[region+ | width | height | filling | decoration]])

e.g., (region [[x | y | width | height | decoration* | filling | header | footer | flow-map?]])

column-set-model: column-set-model type

(define-column-set-model ***variable*** [[column-subset* | fill-out? | tied-column-subset* | filling-direction? | width? | height? | decoration*]]) (column-subset [[column+ | flow-map | top-float-space-below? | bottom-float-space-above? | balance? |justify? | justify-limit? | justify-last-limit? | length-deviation? | length-decrease-order? | align-lines?]]) (column [[width? | height? | x-origin? | y-origin? | footnote-separator? | header? | footer?]])

Annex C
(informative)

Formal public identifiers and their associated characteristics

DSSSL should support to register the formal public identifiers (fpi) for values without description of characteristics. The characteristics are;

— line-breaking-method:

— line-composition-method: .

The characteristics' values should be registered according to ISO/IEC 9070, e.g.,

— "ISO/IEC 10179:1996//LINE COMPOSITION METHOD::KYOTO"

— "ISO/IEC 10179:1996//LINE COMPOSITION METHOD::OXFORD"

— "ISO/IEC 10179:1996//LINE COMPOSITION METHOD::CHICAGO"

— "ISO/IEC 10179:1996//LINE BREAKING METHOD::TRADITIONAL VERTICAL BREAK".