TECHNICAL REPORT

ISO/IEC TR 19758

First edition
2003-04-01

AMENDMENT 1
2005-07-01

Information technology — Document description and processing languages — DSSSL library for complex compositions

AMENDMENT 1: Extensions to basic composition styles and tables

Technologies de l'information — Description de document et langages de traitement — Bibliothèque DSSSL pour compositions complexes —

AMENDEMENT 1: Extensions aux styles de composition de base et aux tableaux

Reference number
ISO/IEC TR 19758:2003/Amd.1:2005(E)

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

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

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.

In exceptional circumstances, the joint technical committee may propose the publication of a Technical Report of one of the following types:

— type 1, when the required support cannot be obtained for the publication of an International Standard, despite repeated efforts;

— type 2, when the subject is still under technical development or where for any other reason there is the future but not immediate possibility of an agreement on an International Standard;

— type 3, when the joint technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Technical Reports of types 1 and 2 are subject to review within three years of publication, to decide whether they can be transformed into International Standards. Technical Reports of type 3 do not necessarily have to be reviewed until the data they provide are considered to be no longer valid or useful.

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 TR 19758:2003 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 34, *Document description and processing languages*.

This amendment provides additional DSSSL library specifications for basic composition styles and table compositions.

Information technology — Document description and processing languages — DSSSL library for complex compositions

AMENDMENT 1
Extensions to basic composition styles and tables

Page 5, Clause 4.4

Add the following clause 4.4.1 before 4.5:

“

4.4.1 Graphical method using the Unwin module concept

Content-driven specification can be determined by using a graphical method based on the Unwin module concept. The following procedure and Figure A1.1 show how all four margins are specified graphically using the Unwin module concept.

a) Specify S point.

b) Draw diagonal line from the top-right edge through S point to bottom left edge.

c) Draw line from left page of bottom-left edge to right page of top-right edge.

d) Draw line from left page of top-left edge to right page of bottom-right edge.

e) Draw line from left page of top-left edge to left page of bottom right edge.

f) Draw line from right page of top-left edge to right page of bottom right edge.

g) Draw horizontal line from S point until reaching line which was drawn in procedure d).

h) Draw vertical line from point A until reaching line which was drawn in procedure b).

i) Draw horizontal line from point B and vertical line from S to set point C.

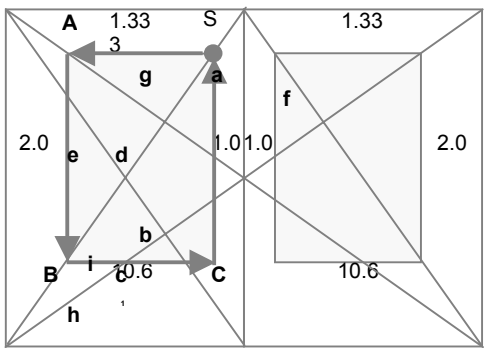

Figure A1.1 — Procedure of graphical method

It is difficult to flow content into areas defined using the Unwin.

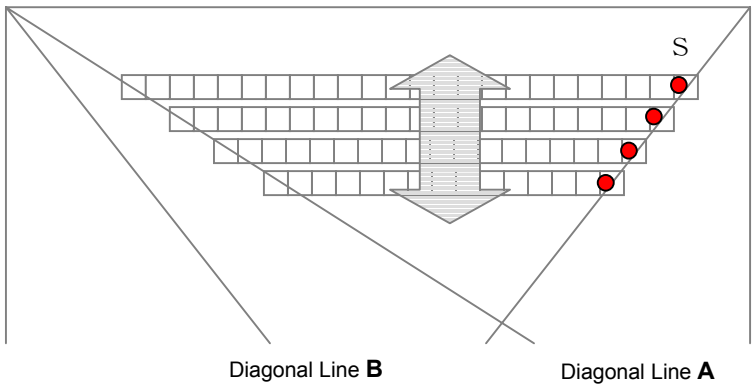

Figure A1.2 — Specification of starting point

The number of em characters in a line and the number of lines in a page must be specified in a content-driven specification. As shown in figure A1.2, starting point S is determined by the closest starting point on diagonal line A that will give the largest number of designated em characters in a line between diagonal lines A and B.

The following equation is used to calculate the length of the line floating direction:

length of line direction = (number of line -1) (line spacing) + (character size)

By adopting this equation, the graphical method can be applied to content-driven method.

When using the above content-driven method as part of a DSSSL library, specification of right margin, font size, number of lines and line spacing is required.

Since the use of four sets of data is not efficient within a DSSSL library, users should limit the inputting data to values of **gutter margin** (*gutter-margin*) and **font size** (*default-font-size*).

The default value of line-spacing is designated as 1.5 em.
”

Page 14, Clause 4.18

Add the following clauses after 4.18.2:

“

4.18.3 Rounded corner table

The shape of a table is rectangular, though corners are allowed to be rounded.

— If you wish to round the corner of table, use *corner-rounded-table-style* under the specification of element table.

— The radius of the corner is specified using (define *base-table-corner-radius* *3pt*). By changing the number within **define**, the radius of table corner is specified as the value of it. The default value of a table corner radius is 3pt.

4.18.4 Table header column and row

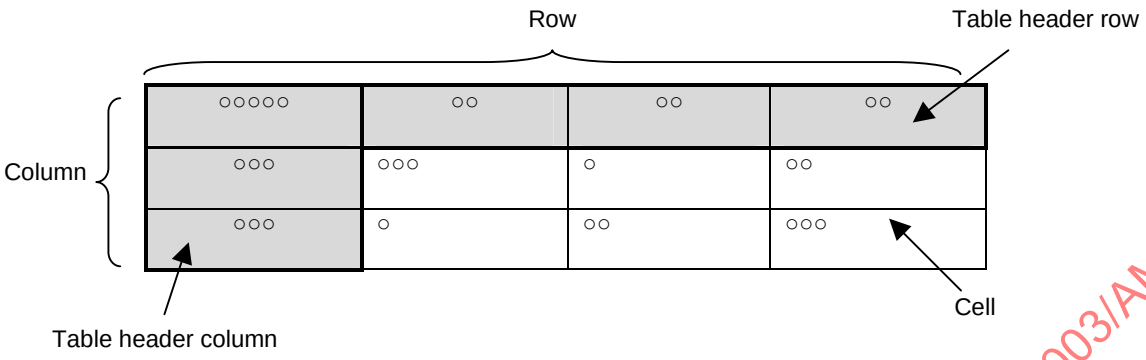

Figure A1.3 — Major Table layout

Table header column <THC> and table header row <THR> should be defined as siblings of table column <TC> and table row <TR> respectively. The table header column and table header row should have the same children as table column and table row respectively, and initially be specified with their background color of light-gray, as shown in figure A1.3.

4.18.5 Multiple strings in table cells

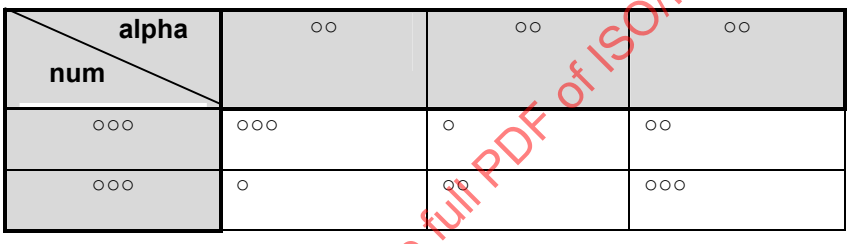

Figure A1.4 — Multiple strings in table cell

Multiple table cell <MTD> should be defined as siblings of table-cell <TD>, and have its children; row string <RSTR> and column string <CSTR>. The row string includes #PCDATA which are related to row data. The column string includes #PCDATA which are related to column data.

The multiple table cell with the following specification will be rendered as shown at the top-left cell of figure A1.4.

```
<MTD>
   <RSTR IMAGESRC="">alpha</RSTR>
   <CSTR IMAGESRC="">num</CSTR>
</MTD>
```

4.18.5.1 Diagonal line through table

If you wish to draw a diagonal line through a whole table, such as a league match table, specify the diagonal line as the attribute of the element TABLE. One restriction of this designation is that the number of columns and rows to which the diagonal is applied must be the same number.

4.18.6 Word wrapping

It is possible to reduce the font size if the string in a cell is larger than the width of the cell. This is designated using the attribute DesFSize="#t", which indicated that the font size is to be reduced if the cell width is exceeded. "#f" means a linefeed must be inserted to wrap the remainder of the word onto a second line.
"

Page 34, Clause 6

Add the following specification before "`;;;;;;;;; Position of image area`":

```
;;;;;;;;;;; graphical method using the Unwin module concept
;;
;;
(define *gutter-margin* 10mm)
(define *default-font-size* 15pt)

(define *a4-width* 210.0mm)
(define *a4-height* 297.0mm)

(define-unit linefeed (* *default-font-size* 1.5))
(declare-initial-value font-size *default-font-size*)
(declare-initial-value line-spacing 1linefeed)

(define _PH_ *a4-height*)
(define _PW_ *a4-width*)
(define _TM_ (* *gutter-margin* (/ _PH_ _PW_)))
(define _BM_ (* 2 _TM_))
(define _LM_ (* 2 _RM_))
(define _RM_ *gutter-margin*)
(define _RW_ (* *default-font-size*
                (ceiling (/ (- *a4-width* (* 3 _RM_)) *default-font-size*))))
(define *linenumber* (ceiling
                 (/ (- (+ (- *a4-height* (* 3 _TM_)) 1linefeed) *default-font-
size*)
                    1linefeed)))
(define _RH_ (+ (* (- *linenumber* 1) 1linefeed) *default-font-size*))

(define-page-model unwin_module_a4_left
  (filling-direction 'top-to-bottom)
  (width _PW_)
  (height _PH_)
  (region
    (width _RW_)
    (height _RH_)
    (x-origin (- *a4-width* _RW_ _RM_))
    (y-origin _BM_))
  )

(define-page-model unwin_module_a4_right
  (filling-direction 'top-to-bottom)
  (width _PW_)
  (height _PH_)
  (region
    (width _RW_)
    (height _RH_)
    (x-origin _RM_)
    (y-origin _BM_))
  )
```

Page 56, Clause 9

Replace existing Table specification with the following specification:

```
;; ============================== TABLES ================================
;; Specification of table corner rounded style
(declare-characteristic table-corner-rounded
  "UNREGISTERED::Next Solution//Characteristic::table-corner-rounded" "#f")

;; Specification of actual table element
(element TABLE
         (let ((number-of-columns
                 (node-list-reduce (node-list-rest (children (current-node)))
                           (lambda (cols nd)
                             (max cols
                                  (node-list-length (children nd))))
                           0)))
         (make display-group
        space-before: %block-sep%
        space-after: %block-sep%
        start-indent: %body-start-indent%
         (with-mode table-caption-mode (process-first-descendant "CAPTION"))
         (make table
                 use: *table-style*
;; Replace below designation to above if you wish to round table corner
;; use: *corner-rounded-table-style*

                 (process-children)))))

(mode table-caption-mode
  (element CAPTION
           (make paragraph
                 ;; use: para-style
                  quadding: 'center
                 font-weight: 'bold
                 space-before: %block-sep%
                 space-after: %para-sep%
                 start-indent: (inherited-start-indent);
                 (literal
                  (string-append
                   "Table "
                   (format-number
                    (element-number) "1") ". "))
                 (process-children-trim))))

(element CAPTION (empty-sosofo)) ; don't show caption inside the table


(element TC (make-column "COLUMN_WIDTH"))
(element THC (make-background-column "COLUMN_WIDTH"))


(element THR
  (make table-row
        use: *background-cell-style*
        (process-children-trim)))
```

```
(element TR
  (make table-row
        (process-children-trim)))

(element MTD
  (make-multiple-cell
   (make paragraph (process-matching-children "RSTR"))
   (make paragraph (process-matching-children "CSTR"))))

(element TH
  (make table-cell
        ;n-rows-spanned: (string->number (attribute-string "COLSPAN"))
        quadding: 'center
        use: *cell-style*
        (make paragraph
              font-weight: 'bold
              space-before: 0.25em
              space-after: 0.25em
              start-indent: 0.25em
              end-indent: 0.25em
              quadding: 'start
              (process-children-trim))))

(element TD
  (make table-cell
        ;n-rows-spanned: (string->number (attribute-string "COLSPAN"))
        quadding: 'center
        use: *cell-style*
        (make paragraph
              space-before: 0.25em
              space-after: 0.25em
              start-indent: 0.25em
              end-indent: 0.25em
              quadding: 'start
              (process-children-trim))))

;; All definition toward table designation
(define *rgb-color-space*
  (color-space "ISO/IEC 10179:1996//Color-Space Family::Device RGB"))
(define (rgb R G B) (color *rgb-color-space* (/ R 255) (/ G 255) (/ B 255)))

(define *lightgray* (rgb 211 211 211))
(define *base-background-color* *lightgray*)
(define *base-table-border* #t)
(define *base-table-corner-radius* 3pt)
(define *base-display-alignment* 'center)
(define *base-cell-margin* 1mm)
(define *base-cell-border* #t)
(define *base-cell-row-alignment* 'center)

(define *table-style*
  (style table-border: *base-table-border*
         display-alignment: *base-display-alignment*))

(define *corner-rounded-table-style*
  (style use: *table-style*
```

```
          table-corner-radius: *base-table-corner-radius*
          table-corner-rounded: "#t"))

(define *cell-margin-style*
  (style cell-before-row-margin: *base-cell-margin*
         cell-after-row-margin: *base-cell-margin*
         cell-before-column-margin: *base-cell-margin*
         cell-after-column-margin: *base-cell-margin*
         ))

(define *cell-border-style*
  (style cell-before-row-border: *base-cell-border*
         cell-before-column-border: *base-cell-border*
         ))

(define *cell-style*
  (style use: (merge-style *cell-border-style* *cell-margin-style*)
         cell-row-alignment: *base-cell-row-alignment*))

(define *background-cell-style*
  (style cell-background?: #t
         background-color: *base-background-color*
         use: *cell-style*
         ))

(define (make-column attribute)
  (make table-column width: (* 1mm (string->number (attribute-string
attribute)))))
(define (make-background-column attribute)
  (make table-column
    use: *background-cell-style*
    width: (* 1mm (string->number (attribute-string attribute)))))

(define *nonborder-style*
  (style cell-before-row-border: #f
         cell-before-column-border: #f))

(define *nonmargin-style*
  (style cell-before-row-margin: 0mm
         cell-after-row-margin: 0mm
         cell-before-column-margin: 0mm
         cell-after-column-margin: 0mm))

(define (unit-columns #!rest ws)
  (apply-map sosofo-append (lambda (w) (make table-column width: (table-unit
w))) ws))

(define *multiple-column-position* 13.5mm)
(define *multiple-row-position* 'center)
(define (make-multiple-pos position)
  (cond ((symbol? position)
         (style quadding: position))
        ((and (quantity? position) (not (real? position)))
         (style quadding: 'start
                start-indent: position))
        (else (error "invalid value for \"make-multiple-pos\"
characteristic"))))
```