DR 11-0008 — WML: Kihon-hanmen-based page design

Status: Further Consideration Required

Subject: WML: Kihon-hanmen-based page design

Qualifier: Request for clarification

Submitter: Murata Organization: JISC

Contact Information: <eb2m-mrt@asahi-net.or.jp>

Submitter’s Defect Number: None

Supporting Document(s): None

Date Circulated by Secretariat: 2011-03-24

Deadline for Response from Editor: 2011-05-24

IS 29500 Reference(s): 29500:2012 Part 1, §17.6.5, “docGrid (Document Grid)”, p. 554

Related DR(s): None

Nature of the Defect:

It is not clear how a program can generate page design based on kihon-hanmen. In other words, how can we compute margin values (attributes of the pgSz element) from kihon-hanmen?

Solution Proposed by the Submitter:

Create a normative appendix in Part 1, and specify the following procedure. (In the final solution, cover multi-column and vertical writing.)

To compute margins from a kihon-hanmen for single-column horizontal documents, the following procedure is recommended:

1. Determine the desired character pitch, say DCP. If it is different from the Normal font size, compute the value of the charSpace attribute.
2. Determine the line pitch, say LP. This is done by adding the line gap and the Normal font size.
3. Let KW= Ch \* DCP \* 20 and KH= Ln \* LP \* 20, where Ch is the number of characters per line and Ln is the number of lines per page.
4. Let RL = pgSz/@w:w - KW and HB = pgSz/@w:h - KH.
5. Set pgSz/@w:left and pgSz/@w:right so that RL = pgSz/@w:left + pgSz/@w:right, and set pgSz/@w:top and pgSz/@w:bottom such that HB = pgSz/@w:top + pgSz/@w:bottom.
6. Set w:docGrid/@w:type="linesAndChars".

Schema Change(s) Needed:

No

**Editor’s Response:**

**2011-09-09 Chris Rae:**

This DR requests that an appendix be added to Part 1, specifying a procedure for converting Kihon-Hanmen layout measurements to the markup required by IS 29500.

I agree that this is a very useful addition to the standard, and I have worked with our WordprocessingML team here to further extend the proposal given by Japan, and cover other scenarios. However, although it is a useful part of the standard, I do not think that this should be a normative annex. It is additional guidance, not documentation of markup, and ought not to affect the compliance of documents. An example of very similar guidance is in Annex I.7, “The Unicode Bidirectional Algorithm and Office Open XML”.

Below are the changes necessary to make a new informative Annex covering Kihon-hanmen - if there is time for national experts to review this before Busan then I'd like to discuss it at the face-to-face.

New annex after Part 1, Annex F

**Annex xxx.   
(informative)  
Kihon-hanmen-based page layout**

Kihon-hanmen is a form of page layout, typically used in East-Asian locales, in which the margins of the page are determined by specifying character size, number of lines per page and number of columns per line. This annex explains a methodology for converting these parameters to the values necessary to effect Kihon-hanmen layout in WordprocessingML.

To specify layout using Kihon-hanmen, the following procedure is suggested:

1. Determine the inputs for the Kihon-hanmen definition:
   1. Determine the desired character size, and specify this as the point size of the Normal font.
   2. If the document has a single column per page:
      1. Determine the desired number of characters per line, and specify this via the value of the charSpace attribute on the docGrid element (using the logic defined in that subclause).
   3. If the document has multiple columns per page:
      1. Determine the number of characters per column using the width of the first column, and specify this via the value of the linePitch attribute on the docGrid element (using the logic defined in that subclause).
   4. Determine the desired number of lines per page, and specify this via the value of the linePitch attribute on the docGrid element (using the logic defined in that subclause).
   5. Set w:docGrid/@w:type to linesAndChars.
2. Determine the resulting width and height of the Kihon-hanmen:
   1. If the document has a single column per page:
      1. Calculate its width in twentieths of a point (KW) as:
   2. If the document has multiple columns per page:
      1. Calculate each column’s width in twentieths of a point as:
      2. Set the values of the cols element appropriately based on the desired column widths, as well as the desired column gaps.
      3. Calculate the Kihon-hanmen’s width in twentieths of a point (KW) as the sum of the column widths + column gaps, using the values of the cols element.
   3. Calculate its height in twentieths of a point (KH) as:
3. Calculate the available margins on the page as follows:
   1. Calculate the remaining width (RL) as:
   2. Calculate the remaining height (HB) as:

If either result is negative, the size of the Kihon-hanmen must be reduced accordingly.

1. Position the Kihon-hanmen on the page as follows:
   1. To center the Kihon-hanmen, set pgSz/@w:left and pgSz/@w:right so that RL = pgSz/@w:left + pgSz/@w:right, and set pgSz/@w:top and pgSz/@w:bottom such that HB = pgSz/@w:top + pgSz/@w:bottom.

To position at a specific location relative to the head of the page, set pgSz/@w:top, and set pgSz/@w:bottom = HB - pgSz/@w:top, and pgSz/@w:left and pgSz/@w:right so that RL = pgSz/@w:left + pgSz/@w:right.

**2011-09-25/28 Busan Meeting:**

Decided to not try and close this for COR2. We request review and comments from East Asian NBs.

**2012-02-06/08, Prague F2F Meeting:**

Although we had a solution for this in Busan, we agreed not to close this DR for COR2, and to request review and comments from East Asian NBs. As no feedback was received, after some discussion, we agreed to ask for feedback once again.

**Action**: Murata-san will ask East Asian NBs for help in resolving DR 11-0008.

**2012-04-15 Prof. Ning Li (via Murata-san):**

I agree with DR-11-0008, DR-11-0009, DR-11-0010 and DR-11-0011 briefly, though there are some minor corrections on the format of text, please see the attachment.

An addition to the above DRs, I have some other comments:

1) In ISO/IEC 29500-1:2011 Office Open XML File Formats — Fundamentals and Markup Language Reference (March 2011), Page 624 said:

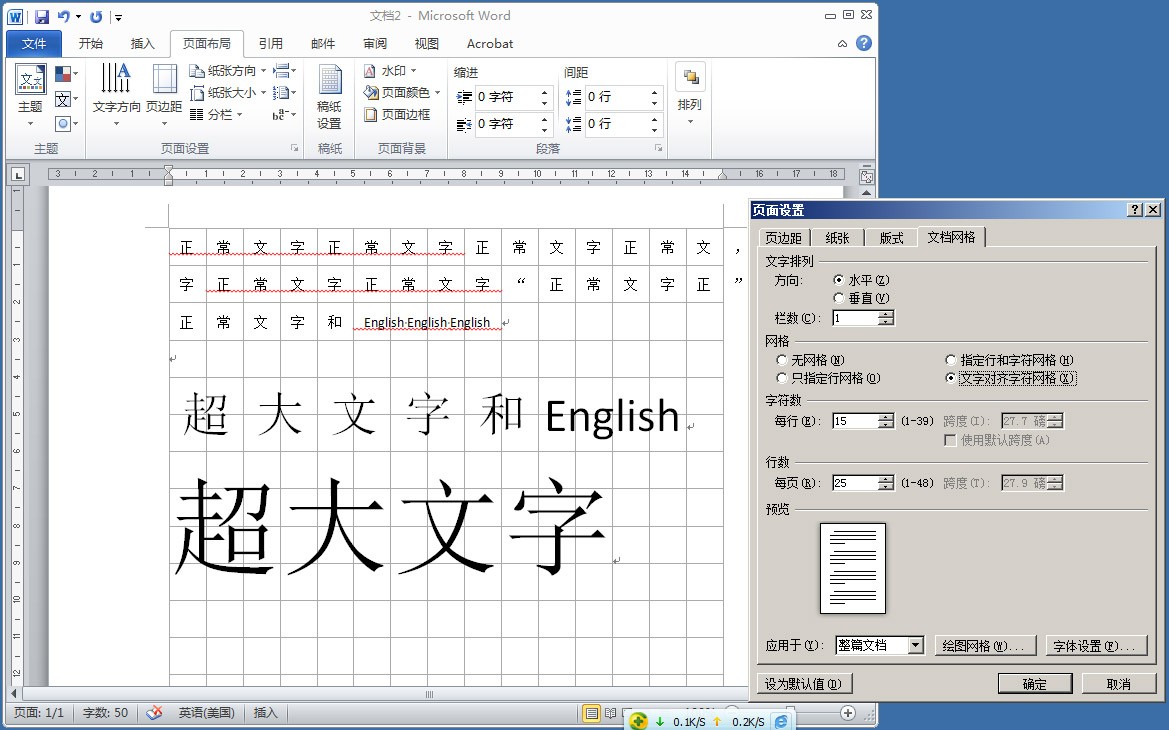
*charSpace(Document Grid Character Pitch) Specifies the number of characters to be allowed on the document grid for each line in this section.*

*…*

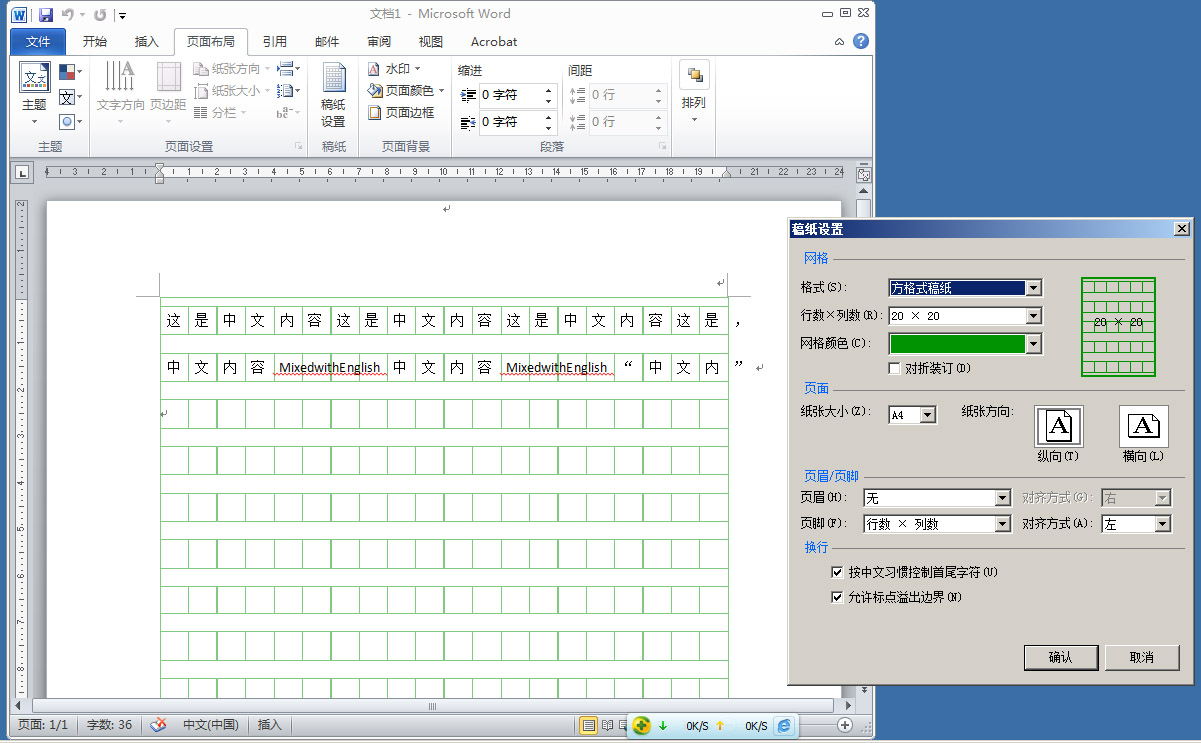
*linePitch (Document Grid Line Pitch) Specifies the number of lines to be allowed on the document grid for the current page assuming all lines have equal line pitch applied to them.*

The words are confusing; at least they are not right definitions.

2) I suggest describe the grid-based layout algorithms in more detail. For example it should specify how to align the characters when the font size is bigger than the grid:



3) A special function for Chinese users (I believe also for Japanese users) is ceijgek. I hope to know: a) the relationship between ceijgek and grid; 2) how to calculate Kihon-hanmen in ceijgek mode; c) again, how to align the characters when the font size is bigger than the grid on ceijgek.



We prefer to treat the grid-based algorithms as the normative part of the standard, as they are crucial to the layout which most of the users of 29500 very much care about.

**2012-05-30 Makoto Murata:**

The Japanese mirror for SC34 reviewed the latest proposal for DR 11-0008. Although we believe that latest proposal is an important progress, we do believe that some important issues have not been addressed. Comments are shown below.

**Kihon-hanmen-based page layout**

Kihon-hanmen is a form of page layout, typically used in East-Asian locales, in which the margins of the page are determined by specifying character size, number of lines per page and number of columns per line. This annex explains a methodology for converting these parameters to the values necessary to effect Kihon-hanmen layout in WordprocessingML.

To specify layout using Kihon-hanmen, the following procedure is suggested:

1. Determine the inputs for the Kihon-hanmen definition:
   1. Determine the desired character size, and specify this as the point size of the Normal font.
   2. If the document has a single column per page:
      1. Determine the desired number of characters per line, and specify this via the value of the charSpace attribute on the docGrid element (using the logic defined in that subclause).
   3. If the document has multiple columns per page:
      1. Determine the number of characters per column using the width of the first column, and specify this via the value of the linePitch attribute on the docGrid element (using the logic defined in that subclause).
   4. Determine the desired number of lines per page, and specify this via the value of the linePitch attribute on the docGrid element (using the logic defined in that subclause).
   5. Set w:docGrid/@w:type to linesAndChars.
2. Determine the resulting width and height of the Kihon-hanmen:
   1. If the document has a single column per page:
      1. Calculate its width in twentieths of a point (KW) as:
   2. If the document has multiple columns per page:
      1. Calculate each column’s width in twentieths of a point as:
      2. Set the values of the cols element appropriately based on the desired column widths, as well as the desired column gaps.
      3. Calculate the Kihon-hanmen’s width in twentieths of a point (KW) as the sum of the column widths + column gaps, using the values of the cols element.
   3. Calculate its height in twentieths of a point (KH) as:
3. Calculate the available margins on the page as follows:
   1. Calculate the remaining width (RL) as:
   2. Calculate the remaining height (HB) as:

If either result is negative, the size of the Kihon-hanmen must be reduced accordingly.

1. Position the Kihon-hanmen on the page as follows:
   1. To center the Kihon-hanmen, set pgSz/@w:left and pgSz/@w:right so that RL = pgSz/@w:left + pgSz/@w:right, and set pgSz/@w:top and pgSz/@w:bottom such that HB = pgSz/@w:top + pgSz/@w:bottom.

To position at a specific location relative to the head of the page, set pgSz/@w:top, and set pgSz/@w:bottom = HB - pgSz/@w:top, and pgSz/@w:left and pgSz/@w:right so that RL = pgSz/@w:left + pgSz/@w:right.

**2012-05-30 Makoto Murata:**

In SC 34/WG4 [N 0239](http://www.itscj.ipsj.or.jp/sc34/wg4/archive/sc34-wg4-2012-0239.pdf) please find a figure depicting our interpretation of the difference between horizontal writing and vertical writing in computing the margin.

**2012-05-31 Teleconference:**

After a short discussion, we agreed to revisit this in Brazil.

**2012-06-25/28, Brasilia F2F Meeting:**

More review of Japan’s comments is needed. There was some talk of having a private teleconference of interested parties to resolve this and then bring the final resolution to WG4. DRs 11-0009, 11-0010, and 11-0011 should be included in the discussion, as they are dependent on this DR.

**2012-09-20 Teleconference:**

We reviewed the history of this DR: After Chris’ initial proposal, CN provided feedback and JP raised a number of questions. JP would like answers to its questions, and we have not yet processed CN’s feedback.

**2015-09-21/24 Beijing Meeting:**

From Murata-san’s email thread, “Comments on the latest proposal for DR 11-0008 — WML: Kihon-hanmen-based page design” from 2015-09-13:

I plan to make real progress in Beijing, although I don't think that we can close this DR there.

First, I now think that we should not provide \*procedures\* for computing margins from Kihon-hanmen or computing Kihon-hanmen from margins.  Rather, we should provide some \*declarative relationships\* among Kihon-hanmen parameters and margin parameters.  This is because there are more than one way to make these parameters consistent.  For example, when character pitches are changed, should we change margins or should we change the number of characters per line?  Both are  
sensible.  Another example is changing line pitches.  Should we then change margins or the number of lines per column?  We do not want to tie the hands of developers by mandating some procedure.

Second, I also think that we should first explain margins, paper sizes, columns, and gaps for horizontal writing and vertical writing. In particular, we should carefully explain which value is physical (i.e., independent from the writing-direction) and which value is logical (i.e., dependent on the writing-direction).  This explanation is useful even when Kihon-hanmen is not used.  Moreover, description of Kihon-hanmen and declarative relationships becomes much easier since we can use column widths and heights rather than margins.

I will further improve my PPTX document, incorporate sketch of proposed changes, and explain them in Beijing.

From Murata-san’s email thread, “Diagrams for understanding Kihon Hanmen or CJK document grid” from 2015-09-14:

Different versions of MS Word handle <w:adjustRightInd/> of WML differently.  MS Word 2013 appears to always adjust the right margin, while MS Word 2007 ajusts it only when @val of this element is true. More about this, see files available at:

<https://www.assembla.com/spaces/IS29500/wiki/Different_interpretations_of_WML_adjustRightIndent>

From Murata-san’s email thread, “Initial draft for addressing CJK layout grid DRs” from 2015-09-22:

Here is a very early draft.

1) Additions to 17.6.3 (Single Column Definition )

When vertical writing is chosen by the value of the val attribute of the textDirection element (17.6.20) for the current section, columns are arranged from top to bottom and the column spacing is vertical.

2) Additions to 17.6.4 (Column Definitions)

If a w:cols element is not present, <w:cols w:equalWidth="1" w:num="1"/> is assumed. (Note: Is this true?)

If a w:cols element does not have the equalWidth attribute, it is assumed to have w:equalWidth="1" (Drafting note: Is this true?)

When vertical writing is chosen by the value of the val attribute of the textDirection element (17.6.20) for the current section, columns are arranged from top to bottom and the column spacing is vertical.

3) Additions to 17.18.14 ST\_DocGrid (Document Grid Types)

Drafting note: Clarify the differences between "linesAndChar" and "snapToGrid". In particular, make clear that "linesAndChar" does not indicate alignment to the character grid.

4) Modifications to 17.3.2.34 snapToGrid (Use Document Grid Settings For Inter-Character Spacing)

Additional character spacing is ALWAYS added.

Alignment to the character grid is guaranteed only when the val attribute of this element is "true" or "1".

5) New subclauses or annex

X.1 Paper size, page margins, and columns

Drafting note: Introduce the first five diagrams and add explain them in prose. Reference 17.6.13 (Page size) and 17.6.11 (Page margins)

X.2 Establishing the Document Grid

Drafting note: Introduce the next four diagrams and explain them in prose. Reference 17.6.5 (Document Grid) and 17.18.14 ST\_DocGrid

(Document Grid Types).

Murata-san presented his document (see N 0320). There was considerable discussion (spread over several days).

**Action**: Murata-san will refine his CJK layout proposal.

**2016-02-29/03-02 Barcelona Meeting:**

Murata-san presented his proposal, as posted in <https://www.assembla.com/spaces/IS29500/documents/d6jO2g23er5zFdacwqjQXA/download/d6jO2g23er5zFdacwqjQXA>.

From Caroline:

After any discussion of this appendix at the face-to-face meeting, I will be happy to offer additional suggestions for improving the English, correcting typos, etc.

For now, I'll point out that the notation for elements and attributes may need explanation.  I don't remember this notation being used elsewhere in Part 1.  Also, references to clauses in the main text (e.g., to §17.6.5 for docGrid) or to locations in the schema may be helpful.

Murata-san responded:

We first have to discuss whether it is a good idea to have one appendix.  We might want to have more than one.  We might want to move the content of the current appendix to existing subclauses.

I heavily used the XPath notation, which has never been used in Part 1.

After some discussion, Murata-san agreed to produce a new version of his proposal, which we’ll then make an official document. It was agreed that most, if not all of this text, should be informative, and perhaps belonged in the Primer annex (after §L.1.7.2, perhaps, and §L.1.7.6), possibly with one or more notes earlier in Part 1 to augment or to point to the new Primer text. At the very least, we should explain somewhere the logical vs. physical distinctions.

**2016-04-12 Teleconference:**

Murata-san needs help w.r.t BIDI, and will contact Darrin re this.

**2016-04-06 Makoto Murata:**

In Barcelona, we agreed that BIDI has to be covered.

In multi-column BIDI documents, the first column is the right column rather than the left column.  This is the biggest difference.

I have updated the document accordingly.  Here goes.

<https://goo.gl/zzOrfQ>

Although wordsmithing is certainly required, I hope to adopt this document and close DR 11-0008 thru -0011 in the next F2F.

**2016-04-06 Makoto Murata:**

In the revised draft, I wrote

"The text direction is right-to-left if a bidi element is present in the current section; it is left-to-right, otherwise.  "

However, it should probably be

"The text direction is right-to-left if a bidi element is present in the current sectPr element; it is left-to-right, otherwise.  "

**2016-06-14/16 Prague Meeting:**

(See Murata-san’s first mail on 2016-04-06.)

Murata-san presented his proposal as revised after the Barcelona meeting. See <https://goo.gl/zzOrfQ>. This was accepted with the edit proposed in his second mail on 2016-04-06.

Issue closed in COR4.

**Action**: Rex will produce the final text of the resolution.

**2016-06-23 Rex Jaeschke:**

**Part 1, §L.1.7.3, “Physical settings and logical settings”, p. xx [New subclause]**

The current section is in horizontal writing when w:sectPr/w:textDirection is absent or the value of w:sectPr/w:textDirection/@w:val is either "tb" or "tbv".The text direction is right-to-left if a bidi element 都市 present in the current sectPr element; it is left-to-right, otherwise. The current section is in vertical writing when the value of w:sectPr/w:textDirection/@w:val is either "rlV", "lrV", "rl", or "lr".

Some page settings are physical: their interpretation does not depend on whether horizontal writing or vertical writing is used. Page size and margin are such absolute settings.

Other page settings are logical: they are interpreted differently depending on whether horizontal writing or vertical writing is used. Column width, column height, and column gap are such logical settings. In horizontal writing, column width, column height, and column gap are horizontal. In vertical writing, they are vertical.

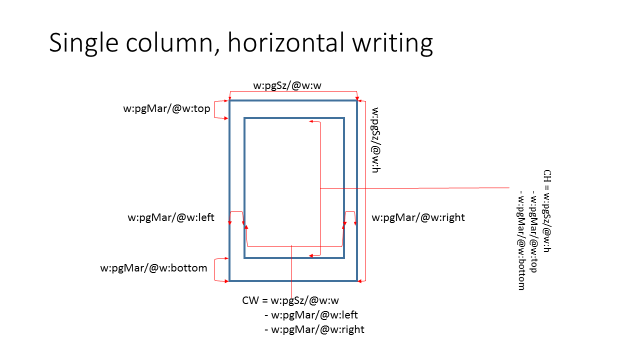
**Part 1, §L.1.7.4. “Margin, paper size, column and gap”, p. xx [New subclause]**

Column width and height are page-layout settings. They do not have XML representations (with the exception of the non-equal-width case). Relevant settings are the page size, page margin, number of columns per page, and column gap. This subclauses summarizes declarative relationships among these page layout settings without specifying procedures for updating them.

Most of the declarative relationships are mathematical equations. These equations are not meant to be exact: the right-hand value and the left-hand value may be slightly different.

This subclause uses a simple notation for referencing XML elements and attributes (See XPath). For example, w:pgSz/@w:w references the value of the w:w attribute of the w:pgSz element in the current section.

**Case 1: Single column, horizontal writing**



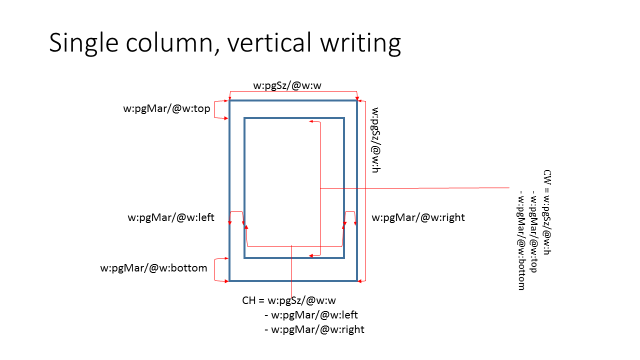
column width =  
 the page width (w:pgSz/@w:w) -  
 - the right page margin (w:pgMar/@w:right)  
 - the left page margin (w:pgMar/@w:left)

column height =  
 the page height (w:pgSz/@w:h)  
 - the top page margin (w:pgMar/@w:top)  
 - the bottom page margin (w:pgMar/@w:bottom)

**Case 2: Single column, horizontal writing, bidirectional**

The same as in Case 1.

**Case 3: Single column, vertical writing**



column width =  
 the page height (w:pgSz/@w:h)  
 - the top page margin (w:pgMar/@w:top)  
 - the bottom page margin (w:pgMar/@w:bottom)

column height =  
 the page width (w:pgSz/@w:w) -  
 - the right page margin (w:pgMar/@w:right)  
 - the left page margin (w:pgMar/@w:left)

**Case 4: Multi-column, horizontal writing (equal width)**



The columns in a page are of the equal width when the value of w:cols/@w:equalWidth is "1" or "true". The number of columns is determined by the value of w:cols/@w:num.

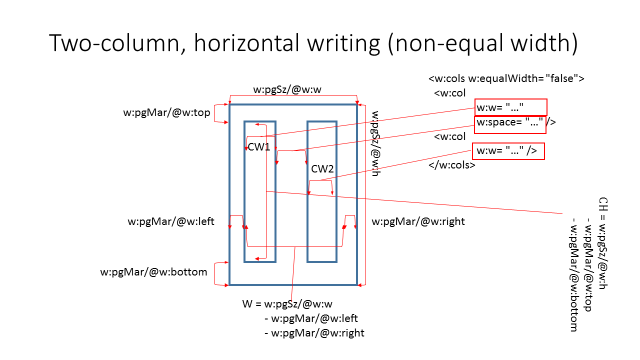
The equation for the column height is the same as in Case 1.

column width =   
 ( the page width (w:pgSz/@w:w)  
 - right page margin (w:pgMar/@w:right) from  
 - left page margin (w:pgMar/@w:left)  
 - w:cols/@w:space \* (w:cols/w:num - 1) )  
 ÷ the number of columns (w:cols/@w:num)

**Case 5: Multi-column, horizontal writing, bidirectional (equal width)**

The same as in Case 4, except that the first column is the right column.

**Case 6 Multi-column, horizontal writing (non-equal width)**

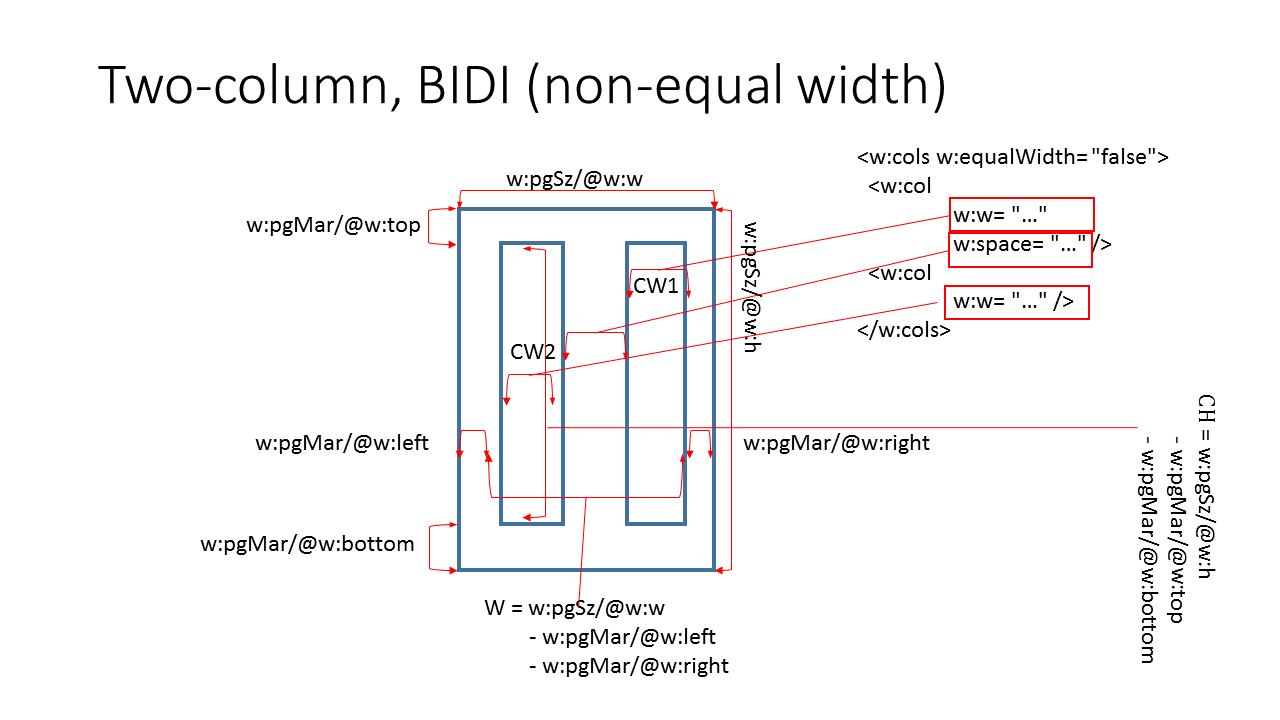


The columns in a page are not of the equal width when the value of w:cols/@w:equalWidth is "0" or "false". The number of columns is determined by the number of the w:cols/w:col child elements.

Every column is of the same height. The equation for the column height is the same as in Case 1.

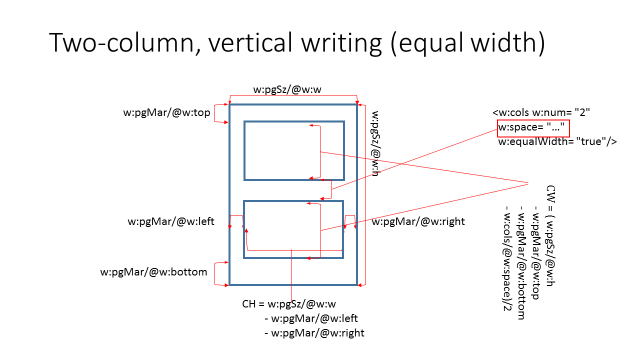
The width of each column (w:col) is specified by the value of its @w:w attribute.

**Case 7: Multi-column, horizontal writing, bidirectional (non-equal width)**



This case is similar to Case 6. But the first column is in the right and the second column is in the left.

**Case 8: Multi-column, vertical writing (equal width)**



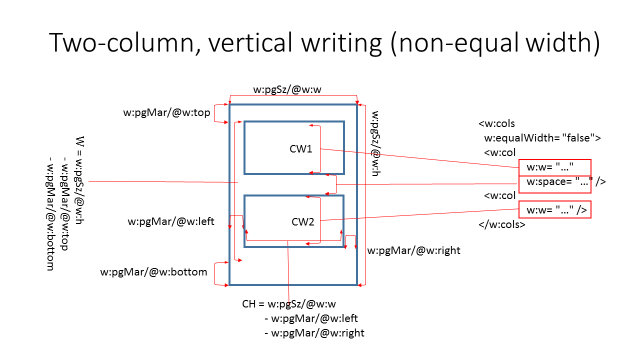
The columns in a page are of the equal height when the value of w:cols/@w:equalWidth is "1" or "true", and the number of columns is determined by the value of w:cols/@w:num.

The equation for the column height is the same as in Case 1.

The equation for the column width is:

column width =  
 (the page height (w:pgSz/@w:h)  
 - the top page margin (w:pgMar/@w:top)  
 - the bottom page margin (w:pgMar/@w:bottom)  
 - w:cols/@w:space \* (w:cols/w:num - 1) )  
 ÷ the number of columns (w:cols/@w:num)

**Case 9: Multi-column, vertical writing (non-equal height)**



As in Case 4, the columns in a page are not of the equal height when the value of w:cols/@w:equalWidth is "0" or "false". The number of columns is determined by the number of the w:cols/w:col child elements.

The equation for the column height is the same as in Case 2.

As in Case 4, the width of each column (w:col) is specified by the value of its @w:w attribute.

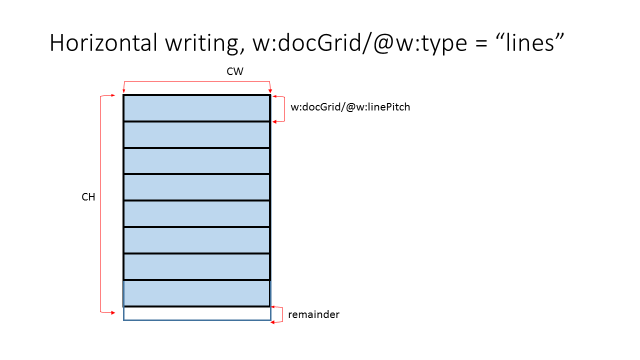
**Part 1, §L.1.7.5. “Document grid”, p. xx [New subclause]**

The docGrid element specifies the settings for the document grid. These settings enables precise layout of full-width East Asian language characters.

When the value of w:docGrid/@w:type is "lines", the docGrid element specifies the number of lines per column, but does not specify the number of characters per line.

When the value of w:docGrid/@w:type is "linesAndChars" or "snapToChars", the docGrid element specifies the number of lines per column as well as the number of characters per line.

**Case 1: Horizontal writing, w:docGrid/@w:type = “lines”**

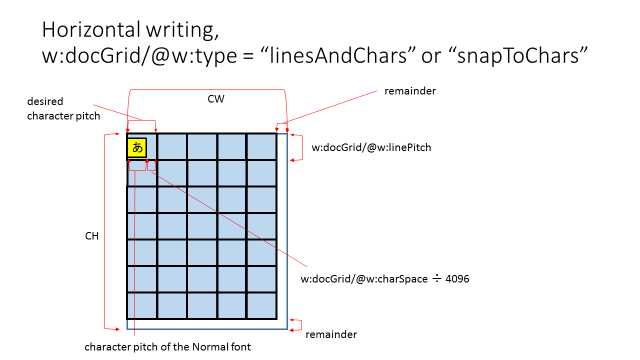


The number of lines per column =  
 the column height ÷ w:docGrid/@w:linePitch

This appendix does not specify which of the column height (and the settings from which the column height is derived), the value of w:docGrid/@w:linePitch, and the number of lines per column should be updated. It merely specifies a declarative relationship among the three settings.

Note: The division operation may lead to non-zero remainder. Confusingly enough, a line of characters may be laid out within the remainder space.

**Case 2: Horizontal writing, w:docGrid/@w:type = “linesAndChars” or “snapToChars”**



The equation for the number of lines per column is the same as in Case 1.

The equation for the numbers of characters per line is:

the numbers of characters per line =  
 the column width ÷ the desired character pitch.

where

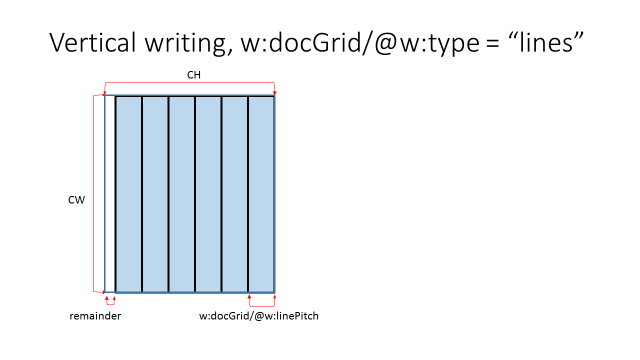
desired character pitch =  
 the character pitch of the Normal font  
 + (w:docGrid/@w:charSpace ÷ 4096)

This annex does not specify which of the column width (and the settings from which the column height is derived), the value of w:docGrid/@w:charSpace, and the number of characters per line should be updated. It merely specifies a declarative relationship among the three settings.

Note: The division operation may lead to non-zero remainder. Then, a character may be laid out within the remainder space.

Note: w:docGrid/@w:charSpace may be negative. Then, the last character does not fit in the line.

**Case 3: Vertical writing, w:docGrid/@w:type = “lines”**

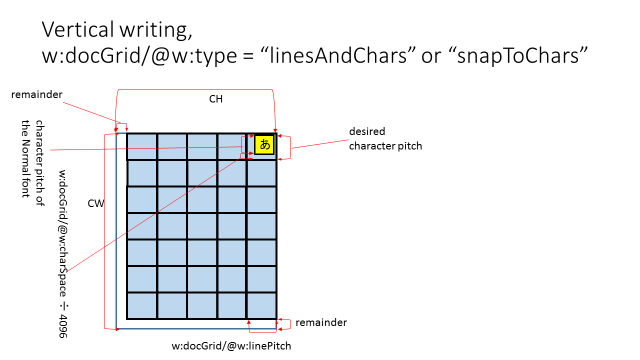


The number of lines per column =  
 the column width ÷ w:docGrid/@w:linePitch

This annex does not specify which of the column width (and the settings from which the column width is derived), the value of w:docGrid/@w:linePitch, and the number of lines per column should be updated. It merely specifies a declarative relationship among the three settings.

Note: As in Case 1, the division operation may lead to non-zero remainder. A line of characters might be laid out within the remainder space.

**Case 4: Vertical writing, w:docGrid/@w:type = “linesAndChars” or “snapToChars”**



The equation for the number of lines per column is the same as in Case 3.

The equation for the number of characters per line is:

the numbers of characters per line  
 = the column width ÷ the desired character pitch.

where

the desired character pitch =  
 the character pitch of the Normal font  
 + (w:docGrid/@w:charSpace ÷ 4096)

Note: As in Case 2, the division operation may lead to non-zero remainder. Then, a character may be laid out within the remainder space.

Note: As in case 2, w:docGrid/@w:charSpace may be negative. Then, the last character does not fit in the line.

**Part 1, §L.1.7.6. “Using the document grid for character formatting”, p. xx [New subclause]**

1) w:docGrid/@w:type = "lines"

This value indicates that line positioning is governed by the document grid but character positioning is not.

Additional line pitch for the alignment with the document grid when <w:pPr> <w:snapToGrid w:val="true" /> </w:pPr>, where "true" is the default.

No additional inter-character spaces for the alignment with the document grid

2) w:docGrid/@w:type = "linesAndChars"

This value indicates that line positioning is governed by the document grid. Character positioning is affected. See JLREQ

*… However, once a kihon-hanmen is established, there is no absolute requirement to align characters with the grid, especially when setting characters inside a line. …*

Additional line pitch for the alignment with the document grid when <w:pPr> <w:snapToGrid w:val="true" /> </w:pPr>, where "true" is the default.

Additional inter-character spaces, but no alignment with the document grid.

3) w:docGrid/@w:type = "snapToChars"

This value indicates that both line and character positioning is governed by the document grid. This value is to simulate manuscript papers, and is not intended for usual documents.

Additional line pitch for the alignment with the document grid when <w:pPr> <w:snapToGrid w:val="true" /> </w:pPr>, where "true" is the default.

Additional inter-character spaces. Complete alignment of CJK characters with the document grid is guaranteed when <w:rPr> <w:snapToGrid w:val= "true" /> </w:rPr>, where "true" is the default.

Changes to Part 1: Y Part 2: N Part 3: N Part 4: N