### ST\_TblLayoutType (Table Layout Type)

This simple [type](type.docx) defines the possible types of layout algorithms which may be used to lay out a [table](table.docx) within a WordprocessingML document.

These algorithms are defined in the following paragraphs (noting, of course, that implementations are free to implement more efficient versions of each).

Fixed Width Table Layout - This method of [table](table.docx) layout uses the preferred widths on the [table](table.docx) items to generate the final sizing of the [table](table.docx), but does not change that size regardless of the contents of each [table](table.docx) cell, hence the [table](table.docx) is fixed width.

[Guidance: Although an application may choose to use a different process, this layout could be performed as follows:

* The [table](table.docx) grid is used to create the set of shared columns in the [table](table.docx) and their initial widths as defined in the [tblGrid](tblGrid.docx) element (§)
* The table’s total width is defined based on the [tblW](tblW.docx) property (§) – if it is set to auto or nil, then the width is not yet determined and will be specified using the row and cell information.
* The first [table](table.docx) row is read and the initial number of grid units before the row starts is skipped. The width of the skipped grid columns is set using the [wBefore](wBefore.docx) property (§).
* The first cell is placed on the grid, and the width of the specified grid column span set by [gridSpan](gridSpan.docx) (§) is set based on the [tcW](tcW.docx) property (§).
* Each additional cell is placed on the grid.
* If at any stage, the preferred width requested for the cells exceeds the preferred width of the [table](table.docx), then each grid column is proportionally reduced in size to fit the [table](table.docx) width.
* If the grid is exceeded (e.g. [tblGrid](tblGrid.docx) specifies three grid columns, but the second cell has a [gridSpan](gridSpan.docx) of three), the grid is dynamically increased with a default width for the new grid column.
* For each subsequent row, cells are placed on the grid, and each grid column is adjusted to be the maximum value of the requested widths (if the widths do not agree) by adding width to the last cell that ends with that grid column. Again, if at any point, the space requested for the cells exceeds the width of the [table](table.docx), then each grid column is proportionally reduced in size to fit the [table](table.docx) width.

end guidance]

The resulting [table](table.docx) shall be displayed regardless of its contents to the size requested.

AutoFit Table Layout - This method of [table](table.docx) layout uses the preferred widths on the [table](table.docx) items to generate the final sizing of the [table](table.docx), but then uses the contents of each cell to determine final column widths.

[Guidance: This layout may be performed in any manner available to an application, but one algorithm as follows may be used:

* Perform the steps above to lay out the fixed width version of the table.
* Calculate the minimum content width - the width of the cell's contents including all possible line breaking locations (or the cell's width, if the width of the content is smaller), and the maximum content width -the width of the cell's contents (assuming no line breaking not generated by explicit line breaks).
* The minimum and maximum content width of all cells that span a single grid column is the minimum and maximum content width of that column.
* For cells which span multiple grid columns, enlarge all cells which it spans as needed to meet that cell's minimum width.
* If any cell in a grid column has a preferred width, the first such width overrides the maximum width of the column's contents.
* Place the text in the cells in the [table](table.docx), respecting the minimum content width of each cell's content. If a cell's minimum content width exceeds the cell's current width, preferences are overridden as follows:
* First, override the column widths by making all other grid columns proportionally smaller until each it at its minimum width. This cell may then grow to any width [between](between.docx) its own minimum and maximum width.
* Next, override the preferred [table](table.docx) width until the [table](table.docx) reaches the page width.
* Finally, force a line break in each cell's contents as needed

end guidance]

[Example: Consider the following fixed width [table](table.docx), which makes extensive use of resized and merged cells on what is actually just a seven-column grid. (The arrows point to each (invisible) vertical line of the grid and the numbers refer to the grid columns):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | |  | |
|  | |  | |
|  |

1 2 3 4 5 6 7

Although the [table](table.docx) is visually complex, the standard rules apply: the first cell in the [table](table.docx) is simply a cell which spans four grid units horizontally, as specified in the [gridSpan](gridSpan.docx) element, and whose preferred with is 2952 twentieths of a point, specified in the [tcW](tcW.docx) element:

<w:[tc](tc.docx)>  
 <w:[tcPr](tcPr.docx)>  
 <w:[tcW](tcW.docx) w:[w](w.docx)="2952" w:[type](type.docx)="dxa"/>  
 <w:[gridSpan](gridSpan.docx) w:val="4"/>  
 </w:[tcPr](tcPr.docx)>  
 <w:[p](p.docx)/>  
</w:[tc](tc.docx)>

Similarly, all cells indented from the stand and end of the grid specify that indent using the [gridBefore](gridBefore.docx) and [gridAfter](gridAfter.docx) elements. For example, the [XML](XML.docx) for the second row in the [table](table.docx) shows that that row starts three grid units into the table:

<w:[tr](tr.docx)>  
 <w:[trPr](trPr.docx)>  
 <w:[gridBefore](gridBefore.docx) w:val="3"/>  
 <w:[wBefore](wBefore.docx) w:[w](w.docx)="2748" w:[type](type.docx)="dxa"/>  
 </w:[trPr](trPr.docx)>  
 …  
</w:[tr](tr.docx)>

If we take this fixed width [table](table.docx) and introduce a long string into the single cell in row three, we see that the presence of this text does not affect cell widths:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | |  | |
|  | |  | |
| longtextstringwithnobreakingcharacters |

If we now turn on the AutoFit property, we see that the algorithm for this AutoFit [table](table.docx) causes grid column two to increase in size, proportionally decreasing the other grid columns’ size to accommodate the long non-breaking string in the last cell:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | |  | |
|  | |  | |
| longtextstringwithnobreakingcharacters |

Each of the other grid columns was reduced, but since all columns are not at their minimum size, the [table](table.docx) width is not increased even though the [table](table.docx) is not yet at the page width. end example]

This simple type's contents are a restriction of the [XML](XML.docx) Schema string datatype.

The following are possible enumeration values for this type:

|  |  |
| --- | --- |
| Enumeration Value | Description |
| autofit (AutoFit Table Layout) | Specifies that this [table](table.docx) shall use an AutoFit [table](table.docx) layout algorithm. |
| fixed (Fixed Width Table Layout) | Specifies that this [table](table.docx) shall use the fixed width [table](table.docx) layout algorithm described above. |

|  |
| --- |
| Referenced By |
| tblLayout@type (§); tblLayout@type (§) |

The following [XML](XML.docx) Schema fragment defines the contents of this simple type:

<simpleType [name](name.docx)="ST\_TblLayoutType">

<restriction base="xsd:string">

<enumeration value="fixed"/>

<enumeration value="autofit"/>

</restriction>

</simpleType>