Elements

Element	Description	Closing Element
<<name>>	Replace this field by whatever data is referenced by "name".	
<<link_name>>	Insert a hyperlink at this location, looking up the URL from the data under the key "name". The URL can optionally specify display text other than the URL to be used with the form: <display text>|<url>. eg "mySite|http://www.my.com"	
<<$abc=name>> <<$abc=10.2>> <<$abc='Fred'>> <<$abc=true>> <<$abc=null>>	Lookup the data associated with "name" and assign it to the variable "abc". Assign the number 10.2 to variable $abc Assign the string "Fred" to variable $abc Assign the boolean true to variable $abc Assign the value null to variable $abc	
<<$abc>>	Lookup the variable "abc" and render its value	
<<cs_name>> <<cs_{expr}>> <<cs_$abc>>	All content up to the closing element is included or excluded depending on the value associated with "name" or the expression "expr" or the variable "abc".	<<es_name>> <<es_{expr}>> <<es_$abc>> <<es_>>
<<rs_name>> <<rs_$abc>> <<rs_people:step2>> <<rs_people:step2down>>	All content up to the closing element is repeated whilst there is data associated with "name" or the variable "abc". "stepN" indicates that the data ("people") should be iterated in steps of N size. When stepping is used, variables $i1, $i2,...$iN are automatically created to access the items available in each step. "stepNdown" indicates that the data ("people") should be iterated in steps of N size and data should be presented in a "down"-ward manner. Variables $i1, $i2,... $iN are automatically created.	<<es_name>> <<es_$abc>> <<es_people:step2>> or <<es_>>
<<cr_name>> <<cr_{expr}>> <<cr_$abc>>	Include the following table rows depending on the value associated with "name" or expression "expr" or the variable "abc".	<<er_name>> <<er_{expr}>> <<er_$abc>> <<er_>>
<<rr_name>> <<rr_$abc>> <<rr_people:step2>>	Repeat the following table rows whilst there is data associated with "name" or the variable "abc". "stepN" indicates that the data ("people") should be iterated in steps of N size. When stepping is used, variables $i1, $i2,...$iN are automatically created to access the items available in each step.	<<er_name>> <<er_$abc>> <<er_>>
<<cc_name>> <<cc_{expr}>> <<cc_$abc>>	Include or exclude the table column containing this field depending on the value associated with "name" or the expression "expr" or the variable "abc".	
Image bookmarked with label or named "img_name" (deprecated "bm_name")	Replace the image with the image data associated with "name" using the default scaling settings (which is stretch). The default setting can be changed by setting the docmosis property docmosis.analyzer.image.scaling.default to fit or stretch. See the Docmosis Developer's Reference for information about setting properties.	
Image bookmarked with label or named "imgstretch_name"	Replace the image with the image data associated with "name" and stretch the image to match the template image placeholder.	
Image bookmarked with label or named "imgfit_name"	Replace the image with the image data associated with "name" and fit the image into the template image placeholder whilst preserving the image aspect ratio.	
<<ref:sub1.doc>>	Insert the template sub1.doc at this location.	
<<refLookup:sub1>>	Lookup sub1 in the data to get the name of the template to insert at this location.	
<<html:myHtml>>	Lookup myHtml in the data and inject as HTML content into the document at this location.	

Expressions

Element	Description
`<<cs_{a<10}>>`	Lookup data element “a” and see if it is **less than** 10 numerically. If “a” is not numeric, a string comparison is performed automatically.
`<<cs_{a='fred'}>>`	Lookup data element “a” and see if it is **equal to** the String literal “fred”.
`<<cs_{$a!=10}>>`	Lookup the variable “a” and see if it is **not equal to** the numeric value 10. If variable “a” does not resolve to a numeric value, a String comparison is performed.
`<<cs_{!hasElements()}>>`	Lookup the data element “`hasElements()`” and **boolean negate** the result. If `hasElements()` returns something other than a boolean, the result will be evaluated using a best effort (eg a String value of “true” would resolve to true). Note that the use of the brackets “()” typically implies the data is to be sourced by calling a Java function literally so it only applies when supplying Java objects as data.
`<<cs_{a=null}>>`	Lookup the data element “a” and determine if it's value is null
`<<cs_{$a}>>`	Determine if the value of the template variable $a is true
Other operators include `<=` `>=`	Less than or equal to Greater than or equal to

Ranges

Element	Description
`<<hotel[0]>>`	The first hotel (indexing starts at zero)
`<<hotel[F]>>`	The first hotel (equivalent to index zero)
`<<hotel[L]>>`	The last hotel
`<<hotel[*]>>`	All hotels
`<<hotel[F3]>>`	The first 3 hotels
`<<hotel[L3]>>`	The Last 3 hotels
`<<hotel[1,2,4]>>`	The hotels at indexes 1,2 and 4
`<<hotel[1-3,L2]>>`	The hotels at indexes 1 to 3 inclusive and the last 2
`<<hotel[0-L2]>>`	All but the last 2 hotels
`<<hotel[3].floor[L].room[0].name>>`	The name of the first room of the last floor of the hotel at index 3

Built-in Variables

Variable	Description
<<$top>> or <<$root>>	The root of the data regardless of the current position or context in the template
<<$this>> or <<$current>>	The current source of data in the current position in the template. This allows for anonymous data lookups from arrays or collections such as <<$current[0]>>.
<<$parent>>	The parent or container of data in the current context of the template. Allows data lookup in the current "hotel" when the current context is a "floor" for example.
<<$idx>>	The current index when iterating through a data set. For example, if we are repeating over all hotels, $idx would report the index of the hotel we are up to. Note that <<$rowidx>> is the same unless using a step size greater than 1.
<<$itemnum>>	Similar to $idx but is the number of the item which we are currently addressing. Item numbering starts at 1. Note that <<$rownum>> is the same unless using a step size greater than 1 .
<<$size>>	The size of the current repeating data set. For example if we are repeating over all hotels, $size would be the number of hotels.
<<$i1>>,<<$i2>>,..<<$iN>>	References to the Nth item when repeating data in "steps of N". For example <<rs_people:step3>> steps through the people in "steps of 3" and Docmosis automatically creates variables $i1, $i2 and $i3 to access each element in the step. For more information about the use of "steps of N" see sections 2.8 Repeating sections (page 24) and 2.9.2 Repeating rows (page 29) of the Developer Guide.
<<$idx1>>,<<$idx2>>,..<<$idxN>>	The absolute indexes of the items when repeating with "steps of N" (as described above) starting at zero.
<<$itemnum1>>,<<$itemnum2>>, ... <<$itemnumN>>	The absolute indexes of the items when repeating with "steps of N" (as described above) starting at one.
<<$rownum>>	The current row number (starting at 1) when repeating (either repeating rows or repeating sections). This is most useful when using the "stepping" directives and the $itemnum is not suitable.
<<$rowidx>>	The current row number (starting at 0) when repeating (either repeating rows or repeating sections). This is most useful when using the "stepping" directives and the $idx is not suitable.