

Docmosis v4.7.5 Release Notes

Oct 2023

New Features / Changes

#	Change
1	Template-Expression engine improved to use BigDecimal for calculations. This avoids the accuracy problems associated with the previous implementation (using doubles). This is transparent and requires no template/code changes.
2	<p>Security update for processing templates to block hidden links (such as OLE object links) from automatically firing when:</p> <ul style="list-style-type: none">a) Storing templatesb) Rendering documentsc) Converting documents <p>The default settings ALLOW all links to fire which is consistent with behaviour before this release. It is expected that templates will rarely contain links that should be allowed to fire so this default system may be changed in a future release.</p> <p>It is recommended that customers test all their templates with the more secure setting:</p> <pre>docmosis.analyzer.preprocessor.externalResource.permission.default=NONE</pre> <p>which makes Docmosis more secure against malicious templates. If problems are detected during testing, a white-list can be established or the templates can be corrected. Then the setting can be left in place.</p> <p>The Java API changes detailed below describe specifics of interacting with the new functionality beyond setting the system-wide defaults.</p> <p>The default setting is to permit ALL by default:</p> <pre>docmosis.analyzer.preprocessor.externalResource.permission.default=ALL</pre>

API Changes

The following API changes should be noted

Class / Interface	Change
DocumentConverter	New method to allow the <code>ExternalResourcePermissions</code> to be specified when converting a document. All existing methods use the default setting (discussed in New Features/Changes above). The new method allows ALL, NONE or a white-list to be specified.
DocumentConversionResult	New method to obtain the links that were in the converted document.
StoreHelper	The permissions (as discussed above) may optionally be specified otherwise the system default is applied.
TemplateAnalysis	New methods to obtain the links in the template and whether the

Class / Interface	Change
	links failed to pass the permissions used when storing.
DocumentProcessor	The permissions (as discussed above) may optionally be specified otherwise the system default is applied.
DropStoreHelper	The permissions (as discussed above) may optionally be specified otherwise the system default is applied.
TemplateSourceMonitor	The permissions (as discussed above) may optionally be specified otherwise the system default is applied.

Bug Fixes / Technical Changes

#	Change
1	Improved processing of SVG images – some images would not parse
2	Added trim() of JSON data added with DataProviderBuilder to allow for padded data

Docmosis v4.7.4 Release Notes

Mar 2023

New Features / Changes

#	Change
1	PDF watermarking improved to allow size, colour, rotation and font to be controlled. Four new methods have been added to <code>ConversionInstruction</code> : <code>setPdfWatermarkColor()</code> <code>setPdfWatermarkRotation()</code> <code>setPdfWatermarkFontSize()</code> <code>setPdfWatermarkFontName()</code> This functionality requires LibreOffice 7.4 or later.
2	Dynamic images will now show in headers/footers of DOCX templates that use the <i>link-to-previous</i> setting. Previously a dynamic image would only show in the first section. Requires LibreOffice 7.4 or later.
3	Security Updates – paths to templates and images are no longer allowed to contain references “upwards” (eg the “../” in “a/ ../b.docx”) so that path references cannot be attempted that could try escaping the expected containment.

API Changes

The following API changes should be noted

Class / Interface	Change
<code>DataProviderFactory</code>	<code>createCompoundDataProvider()</code> method did not work correctly and only one data provider would be added.

Bug Fixes / Technical Changes

#	Change
1	A change in Libre Office 7.5 resulted in numbered lists resetting inside repeating sections. A change to Docmosis has been implemented to provide correct numbering when using LibreOffice 7.5 or later.
2	Stop list numbering (whether dynamic or static) automatically resetting when used in a repeating section when using LibreOffice 7 or later.
3	Stop DocX output adding new unused styles, not referenced in the original template. Requires LibreOffice 7.4 or later.
4	The template function <code>round()</code> has been improved to consistently round “half-up” so that vagaries / inconsistencies with floating point numbers are avoided.
5	Documentation for <code>SystemManager</code> has been improved (developer guide and API Javadoc) to encourage correct usage pattern (which is not to initialise and shutdown for each render).

#	Change
	Docx output has been improved to avoid creation of new styles in the output document. This requires LO 7.4 or later to have any effect.

Docmosis v4.7.3 Release Notes

Sep 2022

New Features / Changes

#	Change
1	Improvements to template-merging <ul style="list-style-type: none">- Static images anchored to last line of sub-template could be lost- Merge could fail with “unable to navigate to bookmark” error when three or more levels of nesting is used and results of document processing was creating blank sub-documents.
2	The template source monitor that can be used to auto-register templates can now register into a specified context. There is a new property to specify the context: <code>docmosis.template.monitor.context=my/context</code>

API Changes

The following API changes should be noted

Class / Interface	Change
TemplateSourceMonitor	TemplateContext can now be specified to load templates into a non-root context

Bug Fixes / Technical Changes

#	Change
1	Fixed issue with launch-time clean up of temporary areas raises an NPE when permissions have been removed from the temporary area.
2	Fix Logging reports unknown classes when bridging to SLF4J/Logback

Docmosis v4.7.2 Release Notes

Jun 2022

New Features / Changes

#	Change
1	Improvement to template-merging to bring “default” style paragraph settings of sub-template into “Standard/Normal” style of sub-template. This fixes issues of sub template styles changing during merging. Specifically, where sub templates with custom styles that are derived from Normal/Standard have implied dependencies on the “default” style.
2	Improvement to template-merging to remove blank space where the rendered sub-template is a blank result.
3	Extend diagnostics logging for object serialization to/from remote converters to assist with in-the field diagnostics.

API Changes

The following API changes should be noted

Class / Interface	Change

Bug Fixes / Technical Changes

#	Change
1	Fixed two template-comments on the same line causing an analysis error that was not helpful.

Docmosis v4.7.1 Release Notes

Apr 2022

New Features / Changes

#	Change

API Changes

The following API changes should be noted

Class / Interface	Change

Bug Fixes / Technical Changes

#	Change
1	Fixed potential remote code exploit in specifically crafted DOCX templates.

Docmosis v4.7.0 Release Notes

Mar 2022

New Features / Changes

#	Change
1	Support for LibreOffice 7.3
2	Support for Java versions up to 17 (minimum Java version remains 6, though LibreOffice 7 requires a minimum of Java 8).
3	<code>log4j</code> use is deprecated in favour of Java util logging.
4	QR codes are now supported (requires the ZXing library to be on the classpath). The implementation is similar to the existing barcode implementation but with some further simplifications.
5	IMB (Intelligent Mail Barcode) format is now supported (requires barcode4j library on the classpath).
6	New template-variables to obtain information about the current template being rendered: <code><<\$templateName>></code> - the name of the template <code><<\$templateFolder>></code> - the folder containing the template <code><<\$templatePath>></code> - both the above combined
7	Barcodes can now use Docmosis template features. This allows barcodes to be dynamically constructed using expressions and variables.
8	A new form of combining templates (" Coordinator Templates ") has been created. This form of combining templates is suitable for document-level type appending, where content is not inserted into a document but instead appended after. This is more appropriate for certain use-cases and reduces complications around headers and footers. The new "coordinator" template type provides "instructions" on how to combine a set of other templates. For PDF, this can "stitch" the final artifacts into one PDF document.



API Changes

The following API changes should be noted

Class / Interface	Change
<code>DocumentRenderResult</code>	This class has been extended to provide more information about the document(s) created. In particular, the template identifier and whether the result is a zip archive. This is important when using the new "coordinator" templates which can return different results than a simple render request.

Bug Fixes / Technical Changes

#	Change
1	A bug in <code>DropStoreHelper</code> has been corrected where it was reloading templates even when they haven't changed.
2	The backward slash (\) character is supported for template paths in addition to the forward slash (/).
3	Document properties can now use <<field>> notation to substitute during document population. This includes custom properties. Note, this functionality is only fully supported with Libre Office version 7.3.
4	Static barcodes (ie defined only by the template and not dependent on any data) now populate even if no data at all is passed to the render process. Previously, some data (even unrelated) had to be present.
5	Template merging – improvements have been made to improve the retention of styles when merging templates without the need for work-arounds.

Docmosis v4.6.3 Release Notes

July 2021

New Features / Changes

#	Change
1	none

API Changes

The following API changes should be noted

Class / Interface	Change

Bug Fixes / Technical Changes

#	Change
1	Improved support for EMF and WMF images.
2	Improved cleanup of temp files when DropStoreHelper.processJarOrZipFile() is used.

Docmosis v4.6.2 Release Notes

May 2021

New Features / Changes

#	Change
1	Updates to template-merging (<code>ref:</code> and <code>refLookup:</code>): <ul style="list-style-type: none">- Manual insertion of a small leading line in the sub template is no longer required in most scenarios- Processing of sub templates cannot override the style in the master/parent templates. This was possible where sub-templates had a "default-font" setting (as opposed to the Normal/Standard style) and LibreOffice versions before 7 are used.
2	The template store now stores the last modified time of the source template when using <code>DropStoreHelper</code> to load templates. This means the <code>DropStoreHelper</code> will also pick up templates that are older than the previously stored version – allowing old versions of a template to replace new versions (eg if a revert is intended).
3	New settings can be applied when generating PDF documents: <code>pdfRestrictPassword</code> <code>pdfRestrictPrinting</code> <code>pdfRestrictEditing</code> <code>pdfRestrictCopy</code> <code>pdfRestrictAllowAccessibilty</code> This is done using the <code>ConversionInstruction</code> class of the Java API.
4	Corrected <code>isBlank()</code> template function to return "true/false" not "1.0/0.0". This incorrect behavior was introduced in Docmosis Java version 4.5.0

API Changes

The following API changes should be noted

Class / Interface	Change
<code>StoreHelper.storeTemplate()</code> where an <code>InputStream</code> is used for the original template	The API now requires the last modified time of the original source template to be specified (since it can't be determined from an <code>InputStream</code>). See feature #2 above for reason.
<code>ConversionInstruction</code>	New methods and constants for specifying PDF output security settings (see feature #3 above).

Bug Fixes / Technical Changes

#	Change
1	Updated text type detection to allow for BOM header
2	Fixed unhelpful error message when processing images from URLs that return a blank image.
3	Fixed a bug where the style of a sub template could affect the style of another sub template in a tree of template-merges due to a miscalculation of unique style names.

#	Change
4	Fixed unhelpful error message when attempting to insert HTML inside a SHAPE object (which is not supported).
5	Tweaks to defend against badly constructed / malicious zip files when processed by DropStoreHelper.

Docmosis v4.6.1 Release Notes

Dec 2020

New Features / Changes

#	Change
1	Sample data now generated for <<html:>> template fields
2	Support for Libre Office 7 now present. Currently LO 6.2.8.2 is still recommended general purpose version.
3	Improvements have been made to processing of large numbers which under Java 15 could be rendered as 1e15 for example rather than a long string of digits.
4	Image substitution from URL data sources better detects error with retrieved image. If invalid, normal error processing will be applied. Previously, the image would be blank.
5	Added a check for a certain type of template pattern that can crash LibreOffice version currently available. Such templates will now be rejected.
6	Allowed the extra-long version of the hyphen (char code 8212) character to be used in field names as a hyphen. This is because Word likes to replace the hyphen with the long or extralong version whilst editing.

API Changes

The following API changes should be noted

Class / Interface	Change

Bug Fixes / Technical Changes

#	Change
1	Improved processing of document styles when merging documents. The sub-template styles are given unique display names.
2	Fixed an issue where merging templates for a multi-output-format render would fail with an error.
3	Fix to delete temp file being left behind when merging templates
4	Improved TemplateStructureExtraction to be able to extract more data-lookup fields from expressions. Expressions using mathematical operators could cause the extraction to stop early leaving some fields undiscovered. This improves the auto/sample data generation.
5	Fixed issue where empty table cell tags could cause a stack overflow error (only occurred in unusual/invalid templates)
6	Fixed issue with some text-format output documents having an invalid byte order mark.
7	Fix issue with artificial sample data elements were generated for Conditional Column (cc_xx) template fields.
8	Fixed issue with char() template function throwing an error if given blank data to process.

#	Change
9	Fixed issue where expressions comparing '1F' and '1D' type of strings would compare them as floating point numbers meaning 1F=1D which is incorrect for strings.
10	Fixed issue with template squote() function raising an error when passed blank/null data.

Docmosis v4.6.0 Release Notes

Apr 2020

New Features / Changes

#	Change
1	Custom Field Renderers now have access to set and get template variables.
2	<code>DropStoreHelper</code> reworked: <ol style="list-style-type: none">1. Deprecated methods removed2. New <code>continueOnError</code> parameter allows continuation or cancellation of processing when errors are found3. New simplified methods for processing URLs and ZIPs4. Will now process ZIP files (in addition to JAR files) when processing URLs or Folders. This may require code changes if deprecated methods were used.
3	Functions that specify locale (such as <code>dateFormat</code> and <code>numFormat</code>) can now specify it also using common notations with language and country. Eg <code>en_US</code> for US English or <code>de_CH</code> for Swiss German.
4	Template function <code>numFormat()</code> has a new additional parameter: <code>formatIsLocalized</code> . It specifies whether the given <code>format</code> string is using characters specific to the given locale, rather than default characters. For example, the default character for year is 'y', but if using the French locale with <code>formatIsLocalized=true</code> , it is 'a'. This parameter defaults to <code>true</code> (preserving current behavior).
5	Template function <code>dateFormat()</code> has two new parameters: <code>ouptutFormatLocalized</code> and <code>inputFormatLocalized</code> which respectively indicate whether the given <code>outputFormat</code> and <code>inputFormat</code> use characters specific to the given <code>outputLocale</code> or <code>inputLocale</code> . For example, the default character for year is 'y', but if using the French locale for output, with matching <code>outputFormatIsLocalized=true</code> , it is 'a'. These parameters both default to <code>false</code> .
6	New template-functions to assist with diagnostics: <code>locale([country or language, name or code])</code> – report the ID for a locale matched with the given parameter <code>localeInfo([country or language, name or code or ID])</code> – report diagnostics for the matched locale <code>localeDatePattern(pattern, locale)</code> – report the given pattern in localized format in the given locale.
7	<code>DocumentConverter</code> now allows a <code>ConversionInstruction</code> to be specified allowing some settings to be applied (passwords, watermarks etc) when converting documents.
8	New template function <code>char(int)</code> which allows characters to be specified by Unicode, html or hexadecimal code.
9	Field parsing has been improved to allow data fields to be optionally specified with brackets around the name and when doing so, hyphenated data names can now be used: <code><<[net-value]>></code> will look for data key "net-value".

#	Change
10	New Template directive <code><<list:reset>></code> can be used to reset the numbering of numbered lists when inside repeating sections. There are use cases where the repeated copies of the list should not continue the numbering from the previous iteration.

API Changes

The following API changes should be noted

Class / Interface	Change
DataProviderBuilder	Removed deprecation for <code>addJavaObject(Object)</code> since it has valid use case and documented warning is sufficient. Added new <code>addJavaObject(Object o, boolean forgiving)</code> . New method <code>addXMLString(InputStream, XMLNodeFilter, boolean)</code> to allow more options for XML data provision
FieldDetails	New methods <code>getTemplateVariable()</code> and <code>setTemplateVariable()</code> to allow interaction with the current state of template-variables during document rendering from customer field renderers.
DropStoreHelper	Reworked – see above.
DocumentConverter	New method <code>convert(File, File, ConversionInstruction)</code> to allow further settings to be specified during conversion.

Bug Fixes / Technical Changes

#	Change
1	Improved processing a corner case in error handling in dev mode not correctly writing the error into a paragraph leading to a corrupted document.
2	Adjusted the behaviour of class loading for Java8 to use the older mechanism. Java 9 onwards using the new mechanism.
3	Improved Locale lookup to be deterministic under different versions of Java – always finding the same locale if available for the same lookup term.
4	Improved conformance to dev-mode vs prod-mode conformance for Java pojo data (reflection) so that rendering will fail if the prod-mode (<code>populationErrorsFatal</code>) when the data is not provided. Previously some of the errors were logged but then suppressed.
5	Improved processing a corner case in error handling in dev mode not correctly writing the error into a paragraph when inside template-comments leading to a corrupted document.
6	XML data provision via <code>DataProviderBuilder</code> using String data no longer uses the platform-default encoding – explicitly uses UTF-8 encoding.
7	Fixed template-expression processing issue where the closing round-bracket character inside quotes was being counted as parameter boundary.

Docmosis v4.5.0 Release Notes

Dec 2019

New Features / Changes

#	Change
1	<p>The Template structure extraction and Dummy Data Providers have been enhanced to extract more information from the fields in the templates. This includes the contents of expression fields (eg <<{firstName + ' ' + lastName}>>, sub-template references etc.</p>
2	<p>New template function dateAdd to allow simple adjustments to dates. For example add a day to myDate:</p> <pre><<{dateAdd(myDate, 1, 'day')}>></pre> <p>subtract 3 weeks from myDate</p> <pre><<{dateAdd(myDate, -3, 'weeks')}>></pre> <p>The units dateAdd allows are millis, seconds, minutes, hours, days, weeks, months, years and can be specified as singular (day) or plural (days).</p> <p>See the Template Guide for more detail.</p>
3	<p>New template function dateDiff to compute the difference between two dates. For example, to compute d2-d1 in days:</p> <pre><<{dateDiff(d1, d2, 'days')}>></pre> <p>The units dateDiff allows are millis, seconds, minutes, hours, days, weeks, months, years and can be specified as singular (day) or plural (days).</p> <p>See the Template Guide for more detail.</p>
4	<p>New template function numToText which will write out numbers in English text. For example:</p> <pre><<{numToText(1024)}>></pre> will output "one thousand and twenty four". <pre><<{numToText(302.24)}>></pre> will output "three hundred and two point two four". <p>See the Template Guide for more detail.</p>

#	Change
5	<p>New template function ordinal which will write out numbers in English text. For example:</p> <pre><<{ordinal(1)}>> will output "1st". <<{ordinal(2, 'suffix')}>> will output "nd". <<{ordinal(23, 'long')}>> will output "twenty third".</pre> <p>See the Template Guide for more detail.</p>
6	<p>New template function numToDollars which will write out numbers in Dollars and cents. For example:</p> <pre><<{numToDollars(1024)}>> will output "one thousand and twenty four dollars". <<{numToDollars(32.24)}>> will output "thirty two dollars and twenty four cents".</pre> <p>See the Template Guide for more detail.</p>
7	<p>New template function mapi() which is the same as the "map" function but ignores case when comparing values.</p> <p>See the Template Guide for more detail.</p>
8	<p>New template function isNumber() will determine whether the input value is a number. This can allow conditional processing based on whether the data is numeric or not. For example:</p> <pre><<cs_{isNumber(invoiceNumber)}>> Process numeric invoice <<else>> Process non-numeric invoice <<es_>></pre> <p>See the Template Guide for more detail.</p>
9	<p>New template-variable: \$nowUTCFormat – provides the date format used for the \$nowUTC value. which allows the new dateAdd() and dateDiff() functions to base calculations on the current date/time.</p>
10	<p>When rendering documents to text format, the output is now UTF-8 format with no "byte order mark".</p>
11	<p>Added support for the FODT document format for use as templates and output documents.</p>
12	<p>Added support for Tables inside shape objects for Word templates.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
TemplateStructureProcessor	<p>This interface requires a new method:</p> <pre>/** * Process a reference to another template * @param ref the template reference details */ public void templateReference (TemplateStructureElement ref);</pre> <p>and implementations will be called when a <<ref:...>> or <<refLookup:...>> field is processed in a template.</p>
TemplateStructureElement	<p>This interface now provides more information about the element being processed:</p> <pre>public boolean isFunctionCall(); public boolean isTemplateReference(); public boolean isTemplateReferenceStatic(); /** * Get any template-expression associated with this field. * * @return null if there is none. */ public TemplateExpression getExpression(); /** * Get the original text of the element from the template */ public String getOriginalFieldText();</pre>
TemplateExpression	<p>A new element providing details about the expression field found in a template.</p>
DataProviderBuilder.addJavaObject(Object o)	<p>This method has been updated to work the same as addJavaObject(Object o, String key) with respect to the property docmosis.populator.lookup.java.forgiving which is detailed in the Javadoc for the method, but it was not being observed.</p>

Bug Fixes / Technical Changes

#	Change
1	Updated authentication system to redirect on success to a calculated url which allows for Load Balancers (with SSL termination) and proxies.
2	Fixed template processing issue with simple empty 1 line nested rs and cs failing to validate
3	Fixed issue where deeply nested 1 line template structure: <<rs_>><<cs_>><<cs_>><<cs_>><<cs_>><<es_>><<es_>><<es_>><<es_>><<es_>> type of structure would fail to analyze.

#	Change
4	Added yyyy-MM-dd'T'HH:mm:ss.SSS'Z' as a default input date format
5	Fixed issue where bottom borders (and probably top borders) were not processing properly when conditional columns were used and repeating rows were meant to dictate the border.
6	Fixed expression processing regarding negation function and functions like isBlank() which would fail to process.
7	Fixed issue with parsing where internal content (soft page breaks) were causing skip sections to overlap for an <<rs_>><<cs_>> on one line scenario and would fail to analyze successfully.
8	Improvements have been made to processing the styles of sub-templates to preserve more of the original sub-template design.
9	Updated image processing to detect the case where an image draw frame (bookmarked for Docmosis) has multiple images inside. This caused corrupt output since it is not expected so now it is raised as an error.
10	Added a step in the sequence of document creation which improves the accuracy of PDF output in some corner cases.
11	Updated image bookmark processing to allow for another quirk of docx processing (SVG tags inside frame and before adjacent bookmark).
12	Detection of SVG placeholder in ODT template causing duplicate images in template with latest versions of LibreOffice. This raises an error in the template during analysis now indicating the problem with the placeholder image.
13	Allow data to specify images from URLs with a second prefix (since the typo is common): [imageURL:...] (in addition to existing [imageUrl:...]).
14	Updated library loading to automatically enable the "custom loader" if Java 8 or later is detected.
15	Fixed issue with Zip processing which could fail to process templates with EMF or WMF images.

Docmosis v4.4.2 Release Notes

Mar 2019

New Features / Changes

#	Change
1	Added XLSX and CSV to known formats and tested auto-feed-into <code>DocumentConverter</code> . This means that Excel files can be converted into CSV.
2	Template structure analysis now includes information about template-references (<code><<ref:xxx>></code> and <code><<refLookup:xxx>></code> fields) which means they can be detected via the

API Changes

The following API changes should be noted

Class / Interface	Change
<code>DataProviderBuilder</code>	<code>addJavaObject(Object)</code> updated to honor the “forgiving” lookup policy when working with Java reflection and Java POJOs. The configured value for <code>"docmosis.populator.lookup.java.forgiving"</code> was being ignored. Improved JavaDoc for <code>addXMLFile(File)</code> to specify how the XML root is included in the data.
<code>TemplateStructureProcessor</code>	A new method signature has been added: <pre>void templateReference(TemplateStructureElement ref);</pre> to allow specific processing to be performed on template references (ref and refLookup fields) in the processed templates.

Bug Fixes / Technical Changes

#	Change
1	fixed corner case regression where an ODT template, with an image bookmarked with a parameterized method that looks up the image would make an additional (redundant and incorrect) bad call for an image on the <code>DataProvider</code> . If the <code>DataProvider</code> was in non-forgiving mode, this would raise an error and fail the render.

Docmosis v4.4.1 Release Notes

Dec 2018

New Features / Changes

#	Change
1	New template functions: <ul style="list-style-type: none">- replaceStr(source, searchFor, replaceWith [, ignoreCase])- replaceFirst(source, searchFor, replaceWith [, ignoreCase])

API Changes

The following API changes should be noted

Class / Interface	Change

Bug Fixes / Technical Changes

#	Change
1	Improved converter bootstrapping process to prime the converter with a more realistic document. This avoids the increased processing time for a first-time-used converter.
2	Fixed issue where table-cell colouring wasn't being applied (when driven by data eg '<bgcolor="#ff0000"/>') to table cell when field providing the style information was number or bullet listed
3	Fixed issue with DOCX image substitution from DOCX templates that could result in the image not being sized correctly and the text 0.00.0 being displayed.
4	Fixed issue where <<cc_>>, <<rr_>>, <<cr_>> tags might be ignored if list-styled (bullet or numbered) in the template.
5	Fixed an issue with TableAnalysis which could mis-track which cell is being processed across the row.
6	Adjusted error handling to not report bad section matches when ending a list if errors have already been reported. This stops two errors being reported for one problem.

Docmosis v4.4.0 Release Notes

Oct 2018

New Features / Changes

#	Change
1	Support for Java up to version 11
2	Support for LibreOffice 6
3	Improved Error Handling in Dev Mode Top level errors now written closer to the location of the error rather than top of document.
4	Improved XML and JSON Sample Data Generation Generated JSON data is now more compact. Sample data generation now supports nested names. For example: <code><<a.b.c>></code> Generates the sample data: <code>{"a": {"b": {"c": "value1"}}</code>
5	New sentenceCase(), isBlank(), ifBlank(), and squote() Functions Use <code>sentenceCase()</code> to convert text data to sentence case format: <code>sentenceCase('this is a sentence.') => This is a sentence.</code> Use <code>isBlank()</code> or <code>ifBlank()</code> to test and act based on blank data values: <code>isBlank('') => true</code> Use <code>squote()</code> to replace double quotes with single quotes in a string literal or data to allow quote-escape: <code>squote('This sentence"s quotes.') => This sentence's quotes.</code>

#	Change
6	<p>New Built-In Template Variables</p> <p>\$num - the current counter into the data.</p> <p>\$num1, \$num2... – provide the absolute index into the repeating sequence starting at zero (i.e. 1,2,3,4,5,6...).</p> <p>\$itemidx1, \$itemidx2... – provide the absolute index into the repeating sequence starting at zero (i.e. 0,1,2,3,4,5...).</p> <p>\$nl - a new line character.</p> <p>\$nowMS – the current UTC time in milliseconds since epoch</p> <p>\$nowUTC - current UTC time as an ISO 8601 string</p> <p>\$quot - a single-quote character</p>
7	<p>Improved Numbered List Processing</p> <p>Now handles more complex cases of:</p> <ul style="list-style-type: none"> - conditioning out some numbered items and resuming numbering - lists continue after sub-lists rendered - allow lists to continue from master template into sub-template using a new "<<list:continue>>" directive

API Changes

The following API changes should be noted

Class / Interface	Change
TemplateStructureElement	New isHyperlink() method determines if the current element is a hyperlink.
TemplateStructureElement	New getContainingNest() method determines if the current element has a nested name.

Bug Fixes / Technical Changes

#	Change
1	Added ability use SSL for remote converters.
2	Improved template-functions to not raise errors when given null values.
3	Improved rendering of merged documents by renaming styles to remove conflict of styles.
4	Updated further functions to handle null/empty params better.
5	Updated remote converter to preserve the extension of the files being transferred. This fixes an issue with HTML injection where the file encoding is not correct if Libre Office imports the file and doesn't realize it is html.
6	Added <<list:continue>> directive to allow sub-templates to specify they want to continue numbering from a master template.

#	Change
7	Improved list processing to track the lists rendered so that stripped lists (e.g. conditional sections) don't break linkages between lists.
8	Fixed bug in TemplateContext constructor which wasn't correctly removing duplicate "/" characters.
9	Fixed bug in SimpleJSONTemplateStructureProcessor.finish() which wasn't keeping the adjustment to the indent variable.
10	Fixed issue with SimpleTemplateTable.hashCode() which was not correctly hashing the array.
11	Updated repeating section processing to correctly handle more complex cases of repeating and variable references. This also fixes a corner case problem with <<pageBreakNotLast>> not working after an internal <<rs_a.b.c>>blah<<es_>> upset the indexing.
12	Fix to writing in-document errors into header and footer (some were hiding).
13	Updated bookmark and image processing to be able to use a bookmark that appears just after the image frame rather than wrapping the image. This helps with imports of Docx files in LO5 and looks like it helps with DOC files where the image has a different wrap mode.
14	Corrected processing of [F-L] type range directives which were doing the wrong thing under some cases.
15	Updated stepping and indexing processes - allows more flexible ranges e.g. [1,2,1,1,2].
16	Improved error messages from template-functions - provide function name in message.
17	Fixed issue resulting in corrupt documents when new line characters in data were mixed with html-like (, <u>, etc.) directives.
18	Fix to HTML injection where <<html:xxx>> fields being last element in a paragraph could leave "repairme" message.

[Older] Docmosis v4.3.0 Release Notes

Nov 2017

New Features / Changes

#	Change
1	<p>Number Formatting more control over Locale</p> <p>The “numFormat” template now supports a new optional parameter to be able to enable / disable the application of the specified Locale when parsing the supplied data:</p> <pre><<{numFormat(value, outputFormat, locale, applyLocaleToInput)}>></pre> <p>The default behavior is to apply the locale to both the input and output, however this is incorrect in some circumstances. For example:</p> <pre><<{numFormat('0.0257', '#.##0,##', 'nl')}>></pre> <p>Incorrectly produces “257” because the “nl” locale is applied when parsing 0.0257 into a number. This can now be corrected by disabling the parse-phase conversion:</p> <pre><<{numFormat(0.0257, '#.##0,##', 'nl', false)}>></pre> <p>So the correct result (“0,03”) is produced.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
DataProviderBuilder	addJSONFile() method now assumes UTF-8 data in the given file.

Bug Fixes / Technical Changes

#	Change
1	Improved template-expression processing: <ul style="list-style-type: none">- processing of unary operators (NOT in particular)- processing of more complex logical expressions - booleans handled better.
2	Improvements to streams dealing with UTF-8 content
3	Improved shutdown process with regards to closing the stderr stream and possible blocking scenario.
4	Improved document processing when merging templates and injecting HTML content to handle more complex cases involving page layout changes and large tables. The result is the layout is

#	Change
	more consistent with the templates involved.
5	Improved issue with LibreOffice 5 installations on Windows platforms where OpenGL processing could cause DocX processing in particular to fail.
6	Improved barcode processing to report errors if barcode is longer than 200 chars.
7	Fix to another corner case of plain text fields not being recognized where expressions were using the ">" greater than operator.
8	Template processing improved with fixes for complex scenarios where errors in templates were unable to be reported and would result in a general failure to process the template instead of correctly indicating the error in the template.
9	Fixed issue where blank "fillin" fields in Word templates were being raised as a general error with no helpful specifics.
10	Fixed template processing error where a table contained repeating rows then a conditional row directive but with no rows inside the condition.
11	Improved "null" error message when a problem occurs processing a template so that implementation can be corrected.

[Older] Docmosis v4.2.0 Release Notes

Jun 2017

New Features / Changes

#	Change
1	<h3>Comments in Templates</h3> <p>The templates can now contain comments. There are two sets of delimiters:</p> <pre><<## and ##>> and <</* and */>></pre> <p>As an example:</p> <pre><<## The following section of the template is regarding xxx ##>> Normal template <<information>></pre> <p>Comments can span multiple lines:</p> <pre><<## The following section of the template is regarding xxx ##>> Normal template <<information>></pre> <p>The two sets of delimiters mean that comments can be nested (one type of comment within another) which can help during development. The following illustrates some normal content with comments being completely commented out by the outer comment delimiters:</p> <pre><</* <<## The following section of the template is regarding xxx ##>> Normal template <<information>> <<## The following section of the template is regarding xxx ##>> Normal template <<information>> */>></pre> <p>See the Docmosis Template Guide for more details.</p>

#	Change
2	<p>Optional Paragraph Fields</p> <p>When a normal template field has no data, the paragraph containing the field remains in the document. For example:</p> <pre data-bbox="240 323 500 415"><<addressLine1>> <<addressLine2>> <<country>></pre> <p>Would create:</p> <pre data-bbox="240 520 422 613">5 Hasler Rd Australia</pre> <p>leaving a blank line if there was no data for “addressLine2”. Optional paragraph fields (identified by the prefix “op:”) remove the containing paragraph if there is no data:</p> <pre data-bbox="240 760 548 852"><<addressLine1>> <<op:addressLine2>> <<country>></pre> <p>So the same data would result in the more desirable:</p> <pre data-bbox="240 957 422 1016">5 Hasler Rd Australia</pre> <p>Optional fields also help with numbered and bullet lists, removing redundant items if the related data is not present. Optional paragraph fields always remove entire paragraphs so can remove multiple lines from documents.</p> <p>See the Docmosis Template Guide for more details.</p>
3	<p>PDF Form Fields can be Pre Filled</p> <p>Templates (ODT format templates only) can create Fillable PDF form fields and pre-populate them. This means that completed or semi completed electronic PDF forms can be generated.</p> <p>See the Docmosis Template Guide for more details.</p>

API Changes

The following API changes should be noted

Class / Interface	Change

Bug Fixes / Technical Changes


#	Change
1	Fixed issue where dynamic images in repeating sections would not render when using versions of Libre Office after version 5.0.
2	Fixed issues where Libre Office versions after 5.0 would not process a template that was open and modified.
3	Improved diagnostics logging for 32-bit / 64-bit mismatch between Java and Libre Office
4	Improved error message when missing section ends are detected
5	Increased plain text default max length from 150 to 1024 characters to support long template expressions
6	Fixed corner-case “null” error when processing complex tables with varying numbers of columns in rows that are repeating.
7	Fixed some character stream outputs which were not writing in UTF-8 explicitly.
8	Improved date parsing to fall-back if performing “Strict” matching and strict matching fails. This provides date parsing that succeeds correctly even when ambiguities are present.
9	Fixed HTML injection issue where inserting blank HTML into a table would leave a “repairme” indicator
10	Improved document loading to explicitly disable macros in templates when loading.

[Older] Docmosis v4.1.0 Release Notes

Aug 2016

New Features / Changes

#	Change
1	<p>Date Rendering has “sticky” timezone.</p> <p>The built in “DateRenderer” and “DateFormat” functions now use the timezone of the input date format (where timezone is provided) to assist with the output formatting. This avoids accidental conversion to a different timezone than the given timezone. Previously the output timezone would be the default for the application environment.</p> <p>For example, if the input data format specifies timezone CST, then the output format implicitly uses CST also.</p> <p>Legacy behavior can be enabled by setting: <code>docmosis.renderer.dateDisableTimezoneDataPersistence=true</code></p>
2	<p>Literals Parsing Improved.</p> <p>Field processing has been enhanced to allow quoted constants to contain arbitrary content. eg ('abc.def'). Previously the period ('.') character would cause the intended literal to be split.</p>
3	<p>Docx Enabled by Default.</p> <p>Docx processing is now enabled by default. This requires LibreOffice which has DOCX support, or the ODFConverter to be used if using Open Office.</p>
4	<p>TIFF Image Support.</p> <p>TIFF images are now supported in the templates and data streams.</p>
5	<p>New toAlpha(), toAlpha2(), toRoman() Functions.</p> <p>New number formatting functions toAlpha, toAlpha2, toRoman have been added to allow Docmosis to support numbering itself. This means that Docmosis can provide numbering in different formats based on the current index:</p> <pre><<rs_items>> <<{toAlpha(\$itemnum)}>>). This is an item <<es_>></pre> <p>Would produce output like:</p> <ul style="list-style-type: none">a). This is an itemb). This is an itemc). This is an itemd). This is an item...

#	Change
	<p><code>toAlpha()</code> maps numbers to a, b, c..., y, z, aa, bb, cc, dd etc. For example:</p> <pre>toAlpha(1) => a toAlpha(26) => z toAlpha(27) => aa toAlpha(28) => bb</pre> <p><code>toAlpha2()</code> maps numbers to a, b, c..., y, z, aa, ab, ac, ad etc. For example:</p> <pre>toAlpha2(1) => a toAlpha2(26) => z toAlpha2(27) => aa toAlpha2(28) => ab</pre> <p>which is the same as <code>toAlpha()</code> except when hitting double letters.</p> <p><code>toRoman()</code> maps numbers to Roman Numerals. For example:</p> <pre>toRoman(1) => i toRoman(26) => ii toRoman(27) => xxvii toRoman(28) => xxviii</pre>
	<p>Barcodes supported by default:</p> <ul style="list-style-type: none"> - Code 128 - Code 39 - ITF 14 <p>Adding a barcode is easy and quite configurable. See the Docmosis Template Guide for details.</p> <p>You will also need <code>barcode4j.jar</code> (from http://barcode4j.sourceforge.net/).</p> 

API Changes

The following API changes should be noted

Class / Interface	Change
<code>ConverterPoolGroupStatus</code>	New public method <code>getUptimeSeconds()</code> to provide the uptime for Docmosis as required.
<code>XMLNodeFilter</code> , <code>StringInterceptor</code> , <code>Base64StringInterceptor</code>	Added to Javadoc API documentation.
<code>DocumentConverter</code>	updated Javadoc to state explicitly the <code>IllegalArgumentException</code> s it throws.

Bug Fixes / Technical Changes

#	Change
1	Fixed bug in <code>FileUtilities</code> where unzipping a zip file with sub folders was failing because the

#	Change
	sub-folders were not being created.
2	Improved field detection where plain text fields near the bottom of a page could be left unrecognized.
3	Updated bootstrapping to work under windows Service accounts.
4	Fixed issue with borders working with <i>stepped</i> repeating rows (ie when using the <code>:stepN</code> directive)
5	Fixed bug where injecting html that was simply a table would leave the "repairme" text behind.
6	Fixed issue where table borders and other style information was lost for some simple tables where <code><<rr >></code> or <code><<ref:>></code> fields existed, but no tables in the document had lookup fields.
7	Reduced diagnostics logging for the <code>ExpressionFunctionAdapter</code> to make sure that when a processing error occurs, the message doesn't contain the package and class of the exception. This makes the message more meaningful to the user.
8	Updated evaluator to not spit out same message about overridden functions every construction.
9	Fixed NPE in table analysis where conditional column was spanned. This could show up as an error with message "null".

[Older] Docmosis v4.0.3 Release Notes

Dec 2015

Please note: html-like markup now defaults to enabled (see `docmosis.populator.field.markup.process` details below).

New Features / Changes

#	Change
1	<h3>If / Else / Else-If Support in Conditional Sections</h3> <p>Simple “else” looks like this:</p> <pre><<cs_true>> true <<else>> false <<es_>></pre> <p>“else-if” is written like this:</p> <pre><<cs_isPerson>> I have a person <<else_isPlant>> I have a plant <<else>> I have something I didn't expect <<es_isPerson>></pre> <p>Conditional Sections Support the new Expression Syntax (described below)</p> <p>Eg:</p> <pre><<cs_{val < 10.0}>> Low value = <<val>> <<else_{val > 100.0}>> High value = <<val>> <<else>> Nominal value = <<val>> <<es_>></pre>
2	<h3>New Expression Engine</h3> <p>A new expression engine has been added that improves computational capabilities of templates. Operators and functions can be applied to String and numeric data. The following lines summarize the way this affects the templates.</p> <p>Expressions in Docmosis templates are still delimited by the braces (“{” and “}”) characters. Expressions can now include:</p> <ul style="list-style-type: none">• Precedence using the brackets “(” and “)” characters• Mathematical expressions• Mathematical and String functions• Boolean logic

#	Change
3	<p>Direct Expression Evaluation</p> <p>Fields which are expressions can be used to insert data into documents. For example:</p> <pre><<{1 + 2 + 3}>> <<{round(val/100,2)}>>% <<{titleCase(firstName + ' ' + lastName)}>></pre>
4	<p>Assignment of Expressions to Variables</p> <p>Variables can now be assigned the results of expressions: << \$m={expr} >></p> <p>Eg. <<\$m={round(1+ceil(2*3.5))}>> round(1+ceil(2*3.5)) = <<\$m>></p>
5	<p>Boolean Logic</p> <pre><<cs_{val1 (val2 && val3)}>> val1 is true or both val2 and val3 are true <<es_>></pre>
6	<p>Maths functions</p> <p>Typical math functions are supported.</p> <p>Eg. <<{max(4.522, 4.5)}>></p> <p>The round() function has been extended to support an optional precision:</p> <pre><<{round(1.236)}>> <<{round(1.236, 2)}>></pre> <p>The precision also pads:</p> <pre><<{round(1.2, 5)}>></pre>
7	<p>String Functions</p> <p>charAt, compareTo, compareToIgnoreCase, concat, endsWith, equals, equalsIgnoreCase, indexOf, lastIndexOf, length, replace (character replacement), startsWith, substring, toLowerCase, toUpperCase, Trim, map, titleCase, split,</p> <p>eg:</p> <pre><<{equals('a', 'b')}>> <<{indexOf('abc', 'b')}>> <<{startsWith('this is', 'this')}>> <<{titleCase('joe blogs')}>></pre>
8	<p>Formatting Functions</p> <p>numFormat and dateFormat functions have been created to perform numeric and date formatting functions. These functions are based on the similarly named FieldRenderers that already existed in Docomsis.</p> <pre>numFormat(<value>, <format>[, <locale>]) <<{numFormat(value1, '###,###.00')}>></pre> <pre>dateFormat(<value>[, <output format>[, <input format>]]) <<{dateFormat(value1, 'dd/MM/yy', 'dd-MMM-yyyy')}>></pre>

#	Change
9	<p>Operators</p> <p>The well known operators are supported: () + - * / % + - = == != < <= > >= && !</p>
10	XML population has been updated to allow "\r" to result in paragraph insertion (in addition to "\r\n" and "\n"). This helps XML data processing where the xml contains this type of new lines.
11	Logging is by default quieter now with more logging information moved to DEBUG/FINE level.
12	Hyperlink processing has been updated to be able to use variables and variables set from expressions. The hyperlinks are now expected to be of the format <<link:xxx>> but the <<link_xxx>> format is still supported.
13	<p>Docmosis can be managed without configuration files. The new <code>Configuration</code> class provides an api for controlling settings:</p> <pre>Configuration config = Configuration.standard(); config.setConverterPoolConfiguration("1"); config.setOfficeLocation(loInstallPathStr); config.setKeyAndSite(siteStr, keyStr); SystemManager.initialise(config);</pre>
14	Sub-templates can now set template-variables that are visible to the "master" template and subsequently processed templates.
15	Updates have been made to allow a load balancer to sit between Docmosis and it's remote converters.
16	String data that is provided is now automatically base64 decoded into images if the string data starts with the sequence "image:base64:"

API Changes

The following API changes should be noted

Class / Interface	Change
<code>Configuration</code>	New <code>Configuration</code> class provides configuration-file-free management of docmosis configuration.
<code>DataProviderBuilder</code>	core - added new methods to <code>DataProviderBuilder</code> to allow "tabular data" to be added easily given a <code>Map[]</code> or a <code>String[][][]</code> .
<code>DataProviderBuilder</code>	New methods for inserting <code>StringInterceptors</code> and getting the list of active interceptors. These interceptors can manipulate data as it is added to the data provider (eg turning base64 image data into binary images).
<code>SystemManager</code>	Exceptions have been simplified for startup. Now the <code>RuntimeException StartupException</code> if a problem occurs during the startup sequence and sub-classes of the <code>StartupException</code> allow problem-specific handling.
<code>RemoteConverter</code> supersedes <code>RemoteConverterTerminus</code>	<code>RemoteConverterTerminus</code> is now superseded by <code>RemoteConverter</code> .

Class / Interface	Change
FieldDetails	The FieldDetails class has a new method getDataProviderLineage() which provides access to the current DataProvider, its parent, grandparent etc. This means broad data access is possible in custom FieldRenderer instances.

Bug Fixes / Technical Changes

#	Change
1	Improved hyperlink insertion to deal with being given blank data. This was corrupting DOC output.
2	Fixed DocumentConverter which was leaking BasicDocuments and relying on finalizer to cleanup
3	Updated LocalOpenOfficeConverter which was hanging onto the inputstreams from spawned docx converters.
4	Updated OpenOfficeServerLauncher to clean up launched Processes better.
5	Updated field parsing to handle more cases where set-variable fields occurring in redundant paragraphs (blank lines) was causing parsing errors.
6	Improved injection of errors into the document (in "dev" mode): <ul style="list-style-type: none"> - Errors not being visible in the document - stopping template errors highlighting in red
7	Improved the performance and accuracy of JSON data in DataProviders. It was converting to/from String unnecessarily AND numeric values (like 100.0) were becoming 100.
8	Improved JSON data provision for anonymous arrays (arrays that have a list of values without keys). Internally, the new key "" is used for items that are anonymous which fits current data provision perfectly - and is semantically a good fit.
9	Improved debug dump toString() of MemoryDataProvider to show blank keys.
10	Improve population of <<rs_ \$this>> and <<rs_ \$current>> to work with JSON anonymous arrays of arrays.
11	Updated analysis to detect when overlapping sections are caused by an over-zealous image bookmark and raise a better error message. This occurs when the bookmark captures more than just the image so other content could be accidentally removed from the result.
12	Updated field processing in numbered and bullet lists. Removes the default assumption that a bullet list with a field in it is intended for repetition and makes it much more "normal".
13	Added configurable diagnostics logging to the DateRenderer
14	Increased the default window size for xml processing for templates with large/complex paragraphs particularly in docx format.
15	Fixed issue where hidden ("_GoBack") bookmarks created by new versions of word (Mac) were causing blank paragraphs to be left behind.
16	Updated default settings to use more memory during template upload (4k to 8k) analyzing templates but improve performance.
17	Updated error handling for when a start or end tag is in a list (but not the matching tag) - causing the related other tag to be lost/remote
18	Updated MemoryDataProvider to delete any created temp files for images when the same

#	Change
	image is set again (overwrite)
19	Updated XML dataprovision to ensure the StringInterceptors are applied for XML attribute values.
20	Fixed issue where looping over including sub-templates more than 2 levels deep would incorrectly determine a cycle in template referencing and raise an error.
21	Updated number renderer to be able to take a second parameter which identifies the locale to use when parsing/formatting the date.
22	Updated default settings to work with newer versions of LibreOffice on Mac OSX.
23	Set the plain text markup << and >> settings as the default so they don't need to be specified in any configuration explicitly.
24	Fixed issue with DataProviderBuilder.addStringInterceptor() methods which only worked for first addition.
25	Added capture and debugging of time spent waiting for the converter at the remote end point to help with performance diagnostics when using remote converters.
26	The license key format has changed.
27	Fixed a corner-case failing to process documents with a conditional section being started in one bullet/list item and being completed in a following paragraph which was not a separate bullet/list item.
28	converterPoolConfig.xml has been simplified: <ul style="list-style-type: none"> - easier to specify "remote" vs "embedded" converters - embedded converters can be launched with a single count="n" setting rather than referring to soffice/soffice.exe n times.
29	HTML-like processing of data values is now enabled by default: <pre>docmosis.populator.field.markup.process=true</pre> whereas it was previously disabled by default. This processing means the ,<i>,<u> and <bgcolor> tags can be embedded in text data to perform some html-like changes. This feature is separate from HTML-injection which is always treated as HTML.
30	The location of the "template store" which is a file-system cache of analysed templates now defaults to "./templateStore". Previously it would default to a temp directory meaning if not set, it would not persist across executions. The default property is now: <pre>docmosis.template.store.location=./templateStore</pre>
31	The converter pool configuration property now defaults to "converterPoolConfig.xml". Previously it had no default and would be typically set to this (now-default) value by all users. The default property is now: <pre>docmosis.document.converter.pool.config.resource=converterPoolConfig.xml</pre>

[Older] Docmosis v3.3.0 Release Notes

Jan 2015

New Features

#	Change
1	<p>New HTML-injection feature. The new tag: <code><<html:myData>></code> will process the given data as html and render the html into the document.</p> <p>So, the following HTML:</p> <pre><p style="border:1px solid orange; width:100%">this is the beginning of html content</p> <h1>This is H1</h1> The heading styles come from the template by default. This H1 heading comes from the template. <h2>This is H2</h2> This H2 is simply a default style from the template also. <h3 style="color:red">This is H3</h3> <p style="width:100%"> We made the above H3 red in the HTML with local style. Interesting. Local styles are important. </p> <table width="100%"><tr> <td style="text-align:center;border:1px solid gray;background-color:#555555">Cell 1</td> <td style="text-align:center;border:1px solid gray;background-color:#555555">Cell 2</td> <td style="text-align:center;border:1px solid gray;background-color:#555555">Cell 3</td> </tr><tr> <td style="border:1px solid gray">And again</td> <td style="border:1px solid gray">more html</td> <td style="border:1px solid gray">and even more</td> </tr><tr> <td style="border:1px solid gray">&nbsp;</td> <td style="border:1px solid gray">10.42</td> <td style="border:1px solid gray">Summary</td> </tr></table> <p>&nbsp;</p> <p style="border:1px solid orange; width:100%">this is the end of html content</p></pre> <p>Will render into a <code><<html:myData>></code> field like this:</p>

#	Change									
	<div style="border: 1px solid orange; padding: 5px;"> <p>this is the beginning of html content</p> <p>This is H1</p> <p>The heading styles come from the template by default. This H1 heading comes from the template.</p> <p>This is H2</p> <p>This H2 is simply a default style from the template also.</p> <p>This is H3</p> <p>We made the above H3 red in the HTML with local style. Interesting. Local styles are important.</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="background-color: #333; color: white;">Cell 1</th> <th style="background-color: #333; color: white;">Cell 2</th> <th style="background-color: #333; color: white;">Cell 3</th> </tr> </thead> <tbody> <tr> <td>And again</td> <td>more html</td> <td style="background-color: #ffcc00;">and even more</td> </tr> <tr> <td></td> <td>10.42</td> <td>Summary</td> </tr> </tbody> </table> <p>this is the end of html content</p> </div>	Cell 1	Cell 2	Cell 3	And again	more html	and even more		10.42	Summary
Cell 1	Cell 2	Cell 3								
And again	more html	and even more								
	10.42	Summary								
2	Image Alt Text can now be dynamically changed. Adding a Docmosis tag to the alt-text of an image will cause document processing to process the tag and update the alt-text.									
3	Accessibility updates for PDF output. To allow PDF documents to be more useful to low-vision users. The API call: <code>ConversionInstruction.setPdfTagged(boolean)</code> causes the PDF result to have extra information (such as alt-text for images) which assist accessibility-tools to present the PDF document to the user.									
4	Improved document handling of document fields –bookmark cross-references will be updated dynamically if they contain dynamic content.									

API Changes

The following API changes should be noted

Class / Interface	Change
ConversionInstruction	New method <code>setPdfTagged(boolean)</code> to add extra information to the PDF output for low-vision assistance.

Bug Fixes / Technical Changes

#	Change
1	Fixed a bug where multiple adjacent set-variable fields were causing an error in template analysis.
2	Fixed a bug where template processing wasn't allowing multiple nested conditional sections on a single line with a Docmosis set-variable field.
3	Improved processing to stop blank lines being left in document after Docmosis content is stripped.

[Older] Docmosis v3.2.0 Release Notes

Sep 2014

New Features

#	Change																	
1	<p>Improvements to the Java download:</p> <ul style="list-style-type: none">- Library loading and class loading- Configuration and property setting <p>to allow docmosis embedded converters to be used by multiple applications deployed into same Web or JEE container.</p> <p>Particularly the new Configuration class allows properties to be set/overridden programmatically.</p>																	
2	<p>New “step down” feature to allow templates to traverse data in steps and “down” first instead of across first. This means data can be written down columns as well as across rows. This applies to both repeating sections (eg “<<rs_items:step3down>>”) and to table rows (eg “<<rr:items:step3down>>”).</p> <p>For example, given a list/array of items [a,b,c,d,e,f,g] the directive <<rr_items:step3Down>> inside a table will create 3 columns of data and populate the first column top to bottom, then move to the second and third columns:</p> <table border="1"><tbody><tr><td>a</td><td>d</td><td>g</td></tr><tr><td>b</td><td>e</td><td></td></tr><tr><td>c</td><td>f</td><td></td></tr></tbody></table> <p>Using the same data, the directive: <<rr_items:step2Down>> inside a table will create 2 columns of data and populate the first column top to bottom, then move to the second:</p> <table border="1"><tbody><tr><td>a</td><td>e</td></tr><tr><td>b</td><td>f</td></tr><tr><td>c</td><td>g</td></tr><tr><td>d</td><td></td></tr></tbody></table> <p>Docmosis will automatically balance the number of rows to fit the given data into the desired number of columns.</p> <p>As with the “stepN” directive, the “stepNdown” directive allocates variables names \$i1, \$i2 ... to allow you to reference data in your array. For a “step3down” directive, \$i1 will be the item for column1, \$i2 for column2 and \$i3 for column3. In a table, the “items” data can be mapped to a 3 column down-first table as follows:</p>	a	d	g	b	e		c	f		a	e	b	f	c	g	d	
a	d	g																
b	e																	
c	f																	
a	e																	
b	f																	
c	g																	
d																		

#	Change																		
	<div data-bbox="354 289 1109 411" style="border: 1px solid black; padding: 5px; margin: 10px auto; width: fit-content;"> <table border="1"> <tr><td colspan="3" style="text-align: left;"><code><<rr_items:step3Down>></code></td></tr> <tr><td style="text-align: left;"><code><<\$i1>></code></td><td style="text-align: left;"><code><<\$i2>></code></td><td style="text-align: left;"><code><<\$i3>></code></td></tr> <tr><td colspan="3" style="text-align: left;"><code><<er_>></code></td></tr> </table> </div> <p data-bbox="224 457 1458 642">Note that \$i1, \$i2 and \$i3 are automatically created by Docmosis to reach the items to be placed into the first, second and third column of the current row. The “items” data may be simple data or they could be structured objects. If they are simple objects, the above template example will render them as expected. If the “items” list contains structured objects, for example person data, then the following shows how the <code>name</code> of the person can be referenced:</p> <div data-bbox="354 680 1109 802" style="border: 1px solid black; padding: 5px; margin: 10px auto; width: fit-content;"> <table border="1"> <tr><td colspan="3" style="text-align: left;"><code><<rr_items:step3Down>></code></td></tr> <tr><td style="text-align: left;"><code><<\$i1.name>></code></td><td style="text-align: left;"><code><<\$i2.name>></code></td><td style="text-align: left;"><code><<\$i3.name>></code></td></tr> <tr><td colspan="3" style="text-align: left;"><code><<er_>></code></td></tr> </table> </div>	<code><<rr_items:step3Down>></code>			<code><<\$i1>></code>	<code><<\$i2>></code>	<code><<\$i3>></code>	<code><<er_>></code>			<code><<rr_items:step3Down>></code>			<code><<\$i1.name>></code>	<code><<\$i2.name>></code>	<code><<\$i3.name>></code>	<code><<er_>></code>		
<code><<rr_items:step3Down>></code>																			
<code><<\$i1>></code>	<code><<\$i2>></code>	<code><<\$i3>></code>																	
<code><<er_>></code>																			
<code><<rr_items:step3Down>></code>																			
<code><<\$i1.name>></code>	<code><<\$i2.name>></code>	<code><<\$i3.name>></code>																	
<code><<er_>></code>																			
3	<p data-bbox="224 856 1466 968">There is a new tag <code><<noRowColoring>></code> (aka <code><<noRowColouring>></code>) which disables the automatic row colouring feature of Docmosis when expanding repeating rows in a table. (sometimes table-row colouring is not desirable).</p> <p data-bbox="224 974 1430 1083">The <code><<noRowColoring>></code> tag can appear in a table to disable it for that table. The tag can also appear in the text body of the document which will disable the row colouring in all following tables.</p>																		
4	<p data-bbox="224 1098 1446 1209">Dates, Booleans and Numeric data passed in as textual information (such as with XML or JSON data) can now be re-formatted by the template. This has been achieved by extending the renderers to also be able to parse data items and even according to specified formats.</p> <p data-bbox="224 1255 480 1287">For example, dates:</p> <p data-bbox="240 1293 1365 1325"><code><<myDate{renderer=date('dd/MMM/yyyy')}>></code> will render myDate into dd/MMM/yyyy</p> <p data-bbox="224 1331 1458 1400">And if your date data is in a special format, you can tell Docmosis how to parse it with a second parameter:</p> <p data-bbox="240 1407 1300 1438"><code><<myDate{renderer=date('dd/MMM/yyyy','EEE MMM dd HH:mm:ss zzz yyyy')}>></code></p> <p data-bbox="224 1486 412 1518">And booleans:</p> <p data-bbox="240 1524 1214 1556"><code><<myItem{renderer=boolean('yn')}>></code> will render myItem into y or n values</p> <p data-bbox="224 1604 412 1635">And numbers:</p> <p data-bbox="240 1642 1312 1673"><code><<myVal{renderer=number('\$00.00')}>></code> will render myVal into \$00.00 style values.</p> <p data-bbox="224 1722 1370 1753">Please see the template guide for more information about the features of the renderers.</p>																		

#	Change
5	<p>Java API and Command Line raw Conversion is now provided by a new API Class: DocumentConverter. This new class does not perform data population based on a template, it is simply to be used to convert between formats (eg ODT->PDF, or DOC->PDF). The DocumentConverter can convert documents at scale using the built in features of Docmosis for scaling document production.</p> <p>A command-line version is provided as shown by example7 in the Docmosis-Java download.</p> <p>Please see the API for more information.</p>
6	<p>Two new built-in variables have been created:</p> <p>\$rownum \$rowidx</p> <p>These items present the current row number (or row index which starts from zero) when repeating data. These new variables are handy when using the “step” functions which affect the \$itemnum and \$idx variables.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
Configuration	New class allowing properties to be set programmatically.
SystemManager	<p>A new method:</p> <pre>initialise(Configuration configuration)</pre> <p>has been added to allow Docmosis to be initialized with programmatic configuration overrides.</p>
DocumentConverter	New class providing direct format conversion rather than document production. Example7 in the Java download contains a command line example of converting files to multiple formats
DateRenderer	The built in date renderer has been extended to parse dates from text data allowing text data to be interpreted and reformatted. Various date formats are applied by default.
BooleanRenderer	The built in boolean renderer has been extended to parse boolean from text data allowing text data to be interpreted and reformatted
NumberRenderer	New built-in renderer that can reformat numeric data, even when that data is provided in text format.

Bug Fixes / Technical Changes

#	Change
1	Fixed issue where unusual configuration could result in infinite loops in data streaming
2	Fixed issue where occasionally a plain text field might not be recognized as a field in a docX template.
3	Fixed issue whereby document-merging was not working with the latest “fresh” build of Libre Office (4.3.0.4). Libre Office is correcting the bug but we corrected for it anyway.

[Older] Docmosis v3.1.0 Release Notes

May 2014

New Features

#	Change
1	<p>Reverse engineering support from Templates</p> <p>The <code>TemplateStructureExtractor</code> and <code>TemplateStructureProcessor</code> classes have been extended to allow sample data to be created for a given template. The list of fields, repeating sections, images etc in a template can be obtained in a descriptive form for programs to use.</p> <p>Implementations available by default:</p> <ul style="list-style-type: none"><code>SimpleXMLTemplateStructureProcessor</code> – describe a template in XML<code>SimpleJSONTemplateStructureProcessor</code> – describe a template in JSON<code>JSONDummyDataTemplateStructureProcessor</code> – create dummy data in JSON format that can populate a template. <p>These classes allow application code to perform various operations based on knowing the structure and fields of a given template including creating forms and creating examples of how data structures should look .</p> <p>See the Java API for more information (http://www.docmosis.com/resources)</p>
2	<p>Template Validation</p> <p>Templates can now be run through the validation mechanism to detect errors without registering them.</p> <p>The new <code>DropStoreHelper.validateTemplate()</code> (also available in <code>StoreHelper</code>) method processes a template to determine if there are any errors and returns a <code>TemplateValidationResult</code> with error messages and suggested fixes.</p> <p>See the Java API for more information (http://www.docmosis.com/resources)</p>
3	<p>“Stepped” Repetition</p> <p>Repeating sections (<code><<rs_>></code>) and repeating table rows (<code><<rr_>></code>) directives can now "step" through data in "chunks". This allows 1-dimensional data (eg a List or Array) to be transposed into groups of 2, 3, 4 etc. For example, an array of images can now be laid out in the document in pairs or triplets etc - controlled by the template and no need to change the structure of the data.</p> <p>See the Template Guide for more details (http://www.docmosis.com/resources)</p>

#	Change
4	<p>Fields can Contain Spaces</p> <p>Space-padded plain text fields are now identified and processed, eg <code><< name >></code> (with spaces around "name") is still recognized as a field. This helps when using copy and paste of text into fields where word processors may add padding.</p> <p>This can be disabled with the property: <code>docmosis.analyzer.field.allowPadded=false</code></p>
5	<p>IndexOf() Supported for Strings, Lists and Arrays</p> <p>The template may use <code><<name.indexOf('dave')>></code> to get have the applicable <code>indexOf()</code> operation invoked for String, List and Array data. For String data, <code>indexOf('dave')</code> returns the index of "dave" within the string data (or -1 if not found). For Lists and Arrays, <code>indexOf()</code> will return the first index of the element equal to "dave" (or -1 if not found).</p>

API Changes

The following API changes should be noted

Class / Interface	Change
RenderRequest	New method to take a template as an <code>InputStream</code> - <code>setTemplateStream()</code>
TemplateStructureProcessor	Added support for creation of data by adding "index" parameter to the <code>repeatBegin()</code> and <code>repeatEnd()</code> methods.

Bug Fixes / Technical Changes

#	Change
1	fixed case where some unmatched section end tags (" <code><<es_</code> ") were not being detected and flagged as an error
2	cleanup of some temporary files being left behind
3	fixed NPE when rendering RTF output on ODT templates with embedded chart objects
4	fixed issue that could cause corrupt document to be produced when trying to write error into document.
5	fixed non-closure of some input streams during processing
6	Fixed possible exception (NPE) raised when having a <code>rr_ / er_</code> structure with no rows between.

[Older] Docmosis v3.0.6 Release Notes

Dec 2013

New Features

#	Change
1	<p>Built in DOCX support option (requires Libre Office rather than OpenOffice)</p> <p>Libre Office 4 DocX support is improving and Docmosis has been upgrade to make some improvements to the DocX result. Docmosis can use Libre Office to produce DocX directly by setting:</p> <pre>docmosis.converter.format.docx.internal.enabled=true</pre> <p>in the <code>docmosis.properties</code> file</p> <p>The <code>odf-converter</code> (third party free system) is the alternative for DocX and is still likely to produce better results in general, but requires a separate program to be installed.</p>
2	<p>Images are now bundled with HTML output</p> <p>When the rendered document contains images and HTML output is chosen, the images are bundled with the result into a ZIP file. This means that when the HTML result is viewed the images are also displayed.</p> <p>When rendering, if only HTML output is selected, then a HTML text result will be returned (ie no images) by default. To provide the images, a ZIP result is required and this can be requested by:</p> <ol style="list-style-type: none">Outputting in more than one format (eg HTML and PDF)Requesting a ZIP for even a single result (the <code>compressSingleFormat</code> flag)
3	<p>Improved Error Messages for Unsupported Output Formats</p> <p>If the underlying office engine (Open Office or Libre Office) doesn't support the requested output format, then this is explicitly detected and stated in the error messages. This is important in cases where one engine supports a format and another doesn't (such as with formats XHTML and DocX).</p>

API Changes

The following API changes should be noted

Class / Interface	Change
<code>DataProviderBuilder</code>	JavaDoc updates for example code

Bug Fixes / Technical Changes

#	Change
1	Added UTF-8 decoding of site key from license to allow multi-byte characters in naming
2	Corrected issue where some error message suggestions were displaying XML (eg <text:s text:c="1"/>)
3	updated to leave templates IN the analyzed stream when info about the image can't be determined. Images were being stripped previously.
4	<p>added new (property-disabled) way to treat templates with spanning-rows with "allow row to break" disabled as an error in the template. This will help where such templates are able to crash Open Office and hence fail to render.</p> <p>The property is disabled by default:</p> <pre>docmosis.analyzer.error.nonBreakableRowSpanningRowsFatal=false</pre> <p>since it is rarely an issue.</p>
5	fixed issue where preserved corrupt file was being cleaned up
6	Updated bundled docmosis.properties file to contain examples of settings where Libre Office 4 is in use.

[Older] Docmosis v3.0.5 Release Notes

Jul 2013

New Features

#	Change
1	<p>Automatic processing of DOCX templates by DropStoreHelper</p> <p>DropStoreHelper previously ignored DocX files when loading templates.</p>
2	<p>API additions to Reverse Engineer from Templates</p> <p>New classes <i>TemplateStructureExtractor</i> and <i>TemplateStructureProcessor</i> provide the ability to interpret the structure of loaded templates and perform arbitrary processing. The <i>SimpleXMLTemplateStructureProcessor</i> is an example implementation that dumps the template structure into an XML format.</p>
3	<p>XML Data Provision Updates</p> <p>When providing data via XML, text data is loaded in a new way to ensure that data that may have an ambiguous meaning is available under both interpretations. For example:</p> <pre><data> <a>some data </data></pre> <p>Will be loaded into the <i>DataProvider</i> such that a field in the template:</p> <pre><<data.a>></pre> <p>will populate with "some data" (this is the new behavior). The following template structure will also populate with "some data":</p> <pre><<rs_data>> <<rs_a>> <<value>> <<es_a>> <<es_data>></pre> <p>(which is the same as the previous behavior). This simply means XML processing should behave more intuitively.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
TemplateStoreFactory	The method <i>getStore(String)</i> can be passed a custom <i>TemplateStore</i> implementation to load. The parameter format is: <i>custom:<impl class>[:param]</i> for example: <i>custom:com.MyStoreImp</i> or <i>custom:com.MyStoreImpl:saveLocation1</i>

Bug Fixes / Technical Changes

#	Change
1	Improved plain text field processing where some fields were not being detected particularly with DOCX format templates.
2	The default example value for the property <i>docmosis.converter.format.docx.external.path</i> in <i>docmosis.properties</i> file has been corrected for linux platforms. It was previously referencing the <i>odf-converter-integrator</i> rather than <i>odf-converter</i> .
3	Javadoc corrected where “wingdings” was being called “windings”.
4	Fixed NPE when <i><bgcolor></i> directive in data was used for <i>docmosis</i> fields that were outside of a table in the template.
5	Fixed a problem where Uploading a template cause cause issues with the converter pool if the Upload crashed OpenOffice

[Older] Docmosis v3.0.4 Release Notes

Feb 2013

New Features

#	Change										
1	Improved Literals Processing Processing of literal values in template fields has been improved. Docmosis can now process numbers, Strings, boolean (true and false) and null values as constants when assigning variables or creating expressions. For example, the following are valid template fields: <table border="1" data-bbox="240 613 1409 852"><tbody><tr><td><<cs_{name='Fred'}>></td><td>test if the value for key "name" is "Fred"</td></tr><tr><td><<cs_{name=null}>></td><td>test if the value for key "name" is null (undefined)</td></tr><tr><td><<cs_{score<10.5}>></td><td>test if the value for key "score" is less than 10.5</td></tr><tr><td><<\$m=false>></td><td>set the template-variable "\$m" to false</td></tr><tr><td><<cs_{\$m=true}>></td><td>test whether the value for template variable "\$m" is true.</td></tr></tbody></table>	<<cs_{name='Fred'}>>	test if the value for key "name" is "Fred"	<<cs_{name=null}>>	test if the value for key "name" is null (undefined)	<<cs_{score<10.5}>>	test if the value for key "score" is less than 10.5	<<\$m=false>>	set the template-variable "\$m" to false	<<cs_{\$m=true}>>	test whether the value for template variable "\$m" is true.
<<cs_{name='Fred'}>>	test if the value for key "name" is "Fred"										
<<cs_{name=null}>>	test if the value for key "name" is null (undefined)										
<<cs_{score<10.5}>>	test if the value for key "score" is less than 10.5										
<<\$m=false>>	set the template-variable "\$m" to false										
<<cs_{\$m=true}>>	test whether the value for template variable "\$m" is true.										

API Changes

The following API changes should be noted
None.

Bug Fixes / Technical Changes

#	Change
1	Added default support for OpenOffice installed on Debian Platforms
2	Added default support for LibreOffice 3.6 on linux platforms
3	Lowered default number of document retries to 1 since the Open Office and Libre Office platforms are stable enough that retrying is rarely required
4	Updated field parsing to handle "smart quotes" created by Word 2010 and later
5	Allow template-variables to be set in the cr_ and rr_ rows of tables (where previously they were quietly ignored). This allows variables to be set in rows that are removed during rendering to be defined and used in subsequent processing.
6	Fix - ensure setting the cell colouring via the data only takes effect if the "docmosis.populator.field.markup.process" property is set to true
7	Improved detection of plain text fields where tag contains a quote (') character
8	Fixed issues where only *.docx named files were running through external DOCX converter (and not *.dotx files)

[Older] Docmosis v3.0.3 Release Notes

May 2012

New Features

#	Change
1	<p>HTML-Like Text Markup</p> <p>The HTML-Like text markup introduced in 3.0.1 has been extended to allow DATA to control the background colour of table cells. If HTML-like markup is active, then any template-field in a table cell can set the background colour by specifying <code><bgcolor="#rrggbb"/></code> as the beginning of the data for the field.</p> <p>For example, given a template field inside a table:</p> <pre><<myName>></pre> <p>If the data contains for key myName contains:</p> <pre><bgcolor="#ff0000"/>James</pre> <p>The field will be populated with the text "James" and the table cell colour will be set to red (#ff0000).</p> <p>The <code><bgcolor></code> tag must be the first item in the data. There doesn't need to be any textual data to match, so you could have the data and the colour in separate fields. As an example the template might look like this:</p> <pre><<myName>><<cellColour>></pre> <p>And the data could contain have:</p> <pre>myName => James cellColour =><bgcolor="#00ff00"/></pre> <p>and the end result would be a green table cell containing the text James.</p>

API Changes

The following API changes should be noted

None.

Bug Fixes / Technical Changes

#	Change
1	Fixed issue where remote converters could be stalled when something other than Docmosis connects and does the correct handshake.
2	Improved plain text markup processing to capture some cases where Docmosis thought the template was invalid.

[Older] Docmosis v3.0.2 Release Notes

April 2012

New Features

None.

API Changes

The following API changes should be noted

Class / Interface	Change
DataProviderBuilder	Improved handing and error reporting for null keys and values.
RenderRequest	Improved javadoc for interaction with ConversionInstruction
ConversionInstruction	Improved javadoc for getConversionFormats() and some PDF-specific methods. Added some new PDF-specific public constants.

Bug Fixes / Technical Changes

#	Change
1	Fixed issue reducing parallel processing using Libre Office. Did not affect Open Office.
2	Improved data processing to correctly compare null and "" (empty string) values between templates and data.
3	Fixed NullPointerException when setting adding null image streams and files and improved messages when null keys used in DataProviderBuilder add() methods.
4	Fixed issue where right-border could disappear when right most columns stripped out of a table using conditional columns and repeating rows. Fixed NullPointerException when a ConversionInstruction passed that has no output formats (ConversionFormat) specified. API documentation improved for ConversionInstruction.
5	Fixed processing to allow RenderRequest settings to work with the contained conversion instruction for determining output formats. API documentation improved for RenderRequest.
6	Updated the shutdown process to provide a grace period for sub-systems to shut down. This fixes spurious errors/warnings during the shutdown process.
7	Updates to some default settings for performance improvements: <ul style="list-style-type: none">- io read block size 4k -> 8k- in memory processing limit 8k -> 16k- worker pool max size 10 -> 20- converter refresh 100 -> 300- working window for analyser 20 -> 40

[Older] Docmosis v3.0.1 Release Notes

March 2012

This release is for most customers a drop-in replacement for the 2.2.2 release. Few parts of the API have incompatible changes, and specific notes are as follows:

1. A new license key is required. License keys for previous versions of Docmosis are not valid. Please visit the Docmosis web site to find out how to obtain your key. If you have purchased a license key within the past 12 months for Docmosis, you will be allowed to upgrade to the new version free of charge.
2. Plain text markup in the templates is turned on by default in the docmosis.properties file that comes with the 3.0.1 release. If you are an existing docmosis user, you will need to add the new properties to your docmosis.properties file if you wish to take advantage of plain text markup. See the new features below for details.
3. HTML-like interpretation of data is disabled by default. This can be enabled in your docmosis.properties file. See the Section below about HTML-like markup.

New Features

#	Change
1	<p>Plain Text Markup</p> <p>Docmosis fields/placeholders can now be written into the document using plain text rather than Open Office Fields or Word merge Fields. This can make template maintenance much simpler.</p> <p>The feature is turned on by default in the downloaded Docmosis bundle. The default delimiters are << and >>. If you are an existing Docmosis user, can add the following properties to your docmosis.properties file:</p> <pre>docmosis.analyzer.field.plainText.prefix=<< docmosis.analyzer.field.plainText.suffix=>></pre> <p>Example1 in the download bundle shows plain text markup in use. It is the same as Example2 except Example1 uses plain text markup and Example2 uses merge fields.</p> <p>Please see the latest <i>Docmosis Template Guide</i> in the Support section of the Docmosis site for more information.</p>
2	<p>Image Scaling Options</p> <p>The template can now indicate three modes of operation when placing images:</p> <ol style="list-style-type: none">1. <i>stretch</i> - images are stretched to fit the template placeholder2. <i>fit</i> - images are scaled to fit the template placeholder but maintaining the original aspect ratio3. <i>default</i> - images will be scaled according to the chosen default behaviour which can be overridden on a per-render basis.

#	Change
	<p>The use of "bm_xxx" to identify an image is now deprecated and replaced by:</p> <ol style="list-style-type: none"> 1. <code>img_xxx</code> - insert the image xxx using the default setting (stretch is the pre-defined default). The stretch/fit behaviour can be changed at the system level using the property: <pre>docmosis.analyzer.image.scaling.default=fit stretch</pre> The behaviour can also be overridden when calling the <code>DocumentProcessor.render(RenderRequest)</code> method. 2. <code>imgstretch_xxx</code> - insert the image xxx stretching the image in x and y directions to fit the template placeholder entirely. 3. <code>imgfit_xxx</code> - insert the image xxx scaling the image to fit the template placeholder, but preserving the aspect ratio of the image. <p>Please see the latest <i>Docmosis Template Guide</i> in the Support section of the Docmosis site for more information.</p>
3	<p>New PDF and Word Controls</p> <p>Various new features can be specified (see API changes below) for PDF or Word output including</p> <ol style="list-style-type: none"> 1. Password Protect 2. Archive Mode and default view for PDF 3. PDF Watermarks <p>Please see the <i>Docmosis Java API</i> or the <i>Docmosis Web Services Guide</i> in the Support section of the Docmosis site for more information.</p>
4	<p>XML and JSON Data Support</p> <p>The engine can now work directly with XML and JSON format data (via the <code>DataProviderBuilder</code> class).</p> <p>Please see the <i>Docmosis Java API</i> in the Support section of the Docmosis site for more information.</p>
5	<p>HTML-like Text Markup</p> <p>String data can now use a limited set of HTML-style mark-up. This allows bold, italic and underline styles to be applied to a single word, phrase or paragraph.</p> <p>For example, your data can contain:</p> <pre>This is bold</pre> <p>Which will be rendered as</p> <pre>This is bold</pre> <p>This is disabled by default and is controlled by the property:</p> <pre>docmosis.populator.field.markup.process=true</pre>

#	Change
	It can be overridden programmatically using the new <code>RenderRequest</code> object for the <code>DocumentProcessor</code> API (see API Changes below). See also <code>example1</code> and <code>example2</code> in the download which show it in operation for the <code><<introduction>></code> placeholder.
6	<p>Experimental DocX Support</p> <p>OpenOffice and LibreOffice currently perform poor conversions to and from docx format files. We have enabled Docmosis to interact with the ODF Converter (http://sourceforge.net/projects/oci/) which performs much better conversions.</p> <p>To use docx support, set the following property in <code>docmosis.properties</code> to indicate where the ODF Converter is installed:</p> <pre>docmosis.converter.format.docx.external.path</pre> <p>eg:</p> <pre>docmosis.converter.format.docx.external.path=c:/program files (x86)/odf-converter-integrator/OdfConverter.exe</pre> <p>You may find it more convenient to install the ODF Converter Integrator (http://katana.oooninja.com/w/odf-converter-integrator) which provides more cross-platform options. Please note the ODF Converter Integrator may reconfigure your host to open docx files with itself rather than Microsoft Word.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
<code>DocumentProcessor</code>	<p>New render method:</p> <pre>render(RenderRequest request)</pre> <p>allowing all and extended features to be set when rendering. It is intended to be the primary method for use in future.</p> <p>The <code>RenderRequest</code> object allows various extra settings including:</p> <ul style="list-style-type: none"> - using multiple Template Stores - overriding default behaviours including <ul style="list-style-type: none"> - image scaling behaviour - error handling behaviour - HTML interpretation of text data
<code>DocumentProcessor</code>	<code>render</code> methods now return a result object which includes the number of pages and the size of the document produced.
<code>ConversionInstruction</code>	<p>Many new setter methods to allow format-specific properties to be controlled including:</p> <ul style="list-style-type: none"> PDF password protect PDF watermarking PDF archive mode (PDF/A-1a) PDF default view settings PDF image compression

	WORD password protect
DataProviderBuilder	New methods to support XML and JSON format data
TemplateIdentifier	A new constructor (String, String) to make it easy to set the name and context at the same time.
ImageScalingDefault	Class defining the constants that can be used for image scaling
TemplateStoreFactory	<p>The method <code>getStore (String)</code> has been removed. A single template store location is expected to be configured using system properties.</p> <p>New methods have been added to allow store instances to be obtained with overriding settings for:</p> <ul style="list-style-type: none"> - whether template errors are fatal (loading into the store results in an exception) or allowed (templates are stored even with errors which can then be shown when rendered) - delimiters to use when analysing templates

Bug Fixes / Technical Changes

#	Change
1	Improved processing and reporting of errors in template tables where rows marked as heading rows are used.
2	Improved handling of <code>TemplateIdentifiers</code> and <code>TemplateContexts</code> to ensure <code>Engine</code> , <code>Cache</code> and <code>DropStoreHelper</code> work consistently to identify a template. This corrects a problem where a template update would not be dynamically picked up when rendering.
3	Fixed issue when rendering in multiple formats at the same time - images may not be rendered into final document.
4	Fixed issue where "File In Use" could occur when Update a template on the fly
5	Field Renderers now can be applied in Headers and Footers
6	Fixed NPE that could occur when including an external template and the current template context was "" (empty).
7	Community edition limited to 200 documents per day
8	Improved processing of consecutive spaces which were sometimes condensed to a single space.
9	A <code>refLookup</code> that could not find the template name could result in an error that was not meaningful.

[Older] Docmosis v2.2.2 Release Notes

January 2011

New Features

#	Change
1	<p>New Break Fields</p> <p>Docmosis now provides 4 fields as a neater way to insert page and column breaks. The first forms will produce a page and column break (for multi-column documents) respectively:</p> <p><i>«pageBreak»</i> <i>«columnBreak»</i></p> <p>The second forms are for use within repeating sections and will insert the corresponding break until the last iteration of the section:</p> <p><i>«pageBreakNotLast»</i> <i>«columnBreakNotLast»</i></p> <p>The new fields can be used instead of or in conjunction with actual page and column breaks in your template. Please see example4 in the download bundle which shows an example use of <i>«pageBreakNotLast»</i>.</p>
2	<p>Image Lookup Extended</p> <p>Image lookup has been extended to match the facilities of textual data including nesting (eg my.images.image1) and variables (\$myImage)</p>
3	<p>Corrupt Templates are Detected Earlier</p> <p>Previously if badly corrupted documents were used as templates (for example empty files or incorrect format) then Docmosis would take minutes to identify this. Now such templates are recognised sub-second.</p>
4	<p>Full-Justification Support for Multi-Line Data</p> <p>Carriage returns/end-of-line sequences in data now result in the end of a paragraph in the resulting document. This is particularly useful when using full-justified paragraph formats which would previously display overly-spaced lines.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
DataProviderBuilder	New addJavaObject(Object, String, boolean) method has been added to allow "forgiving" mode to be specified during Java reflection for data.
DataProviderBuilder	addJavaObject(Object) has been deprecated since it can mask other data and such behaviour is not obvious.

Bug Fixes / Technical Enhancements

#	Change
1	Improved processing and reporting of errors in template tables where rows marked as heading rows are used.
2	Corrected logging of class name where Reflection fails to find any method related to the template field.
3	Fixed issue where images were not substituting in headers under some conditions

[Older] Docmosis v2.2.1 Release Notes

May 2010

New Features

None

API Changes

The following API changes should be noted

Class / Interface	Change
DataProviderBuilder.addJavaObject(Object)	This method has been documented as dangerous due to its ability to hide other data. addJavaObject(Object,String) is the safe alternative.

Bug Fixes / Technical Enhancements

#	Change
1	Reflection now populates correctly from non-List Collections such as Set (eg TreeSet) and Queue.

[Older] Docmosis v2.2.0 Release Notes

April 2010

New Features

#	Change
1	<p>Template Merging</p> <p>Docmosis now allows templates to be referenced by other templates and the templates will be merged at processing time. This allows templates to have common content to be separated out into shared templates.</p> <p>For example, if documents have a standard header layout, they can reference a common header template using one of the two new Docmosis fields. This might look like:</p> <p>«ref:header.doc» or «refLookup:headerTemplate»</p> <p>where the first field would pull in the template called <i>header.doc</i> into the current template and the second field would look up the key “<i>headerTemplate</i>” in the data provider to get the name of the template to pull in. See the Docmosis Template Guide for more information and <i>example5</i> in the download bundle for an example.</p>
2	<p>New Render Methods</p> <p>The DocumentProcess class has new overloaded render methods to assist in simplifying document generation. Most notably is the presence of a new Boolean parameter to override the default behaviour of cleaning up the DataProvider after the render. Reusing the data can be helpful if making separate render calls to produce separate documents rather than a single call to produce a zip archive.</p>
3	<p>Reduced IO</p> <p>Disk and Network IO has been reduced in the case where the Converters are on the same host as the core engine.</p>
4	<p>Improved Error and Diagnostic Handling</p> <ul style="list-style-type: none">a) 32/64 bit incompatibilities are detected and suggestions are reported. This is helpful when using a 64 bit Java and there is no 64 bit OpenOffice (such as on Windows and MacOSX).b) Handshaking between the Docmosis core and the converters will report versioning issues, status and environmental differences.
5	<p>Improved docmosis.properties</p> <p>The example <i>docmosis.properties</i> file has been re-organised to show the critical properties first, better documentation and show some of the other important properties.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
-------------------	--------

DocumentProcessor	New renderDoc() methods allowing easier access to re-using the same DataProvider between calls. Previously only one method allowed this to be specified (within the ConversionInstruction)
-------------------	--

Bug Fixes / Technical Enhancements

#	Change
1	Repeating Table Rows now allow for conditional rows and hence keep border and background styling features
2	Improvements to rendering errors into the resulting document in corner cases where error was not reported
3	Improvements to ODT (OpenOffice Writer) template processing for end of section detection.
4	Improvements to JavaDoc information

[Older] Docmosis v2.1.1 Release Notes

February 2010

Bug Fixes / Technical Enhancements

#	Change
1	<p>OpenOffice 3.2 support</p> <p>OpenOffice has a great new release which has no serious bugs as far as Docmosis is concerned. The previous two production releases (3.0.1 and 3.1.0) had bugs that were not ideal for typical Docmosis use.</p> <p>OpenOffice 3.2 changes a few things under the hood so Docmosis had to upgrade to match.</p>
2	<p>New options to control the way Docmosis loads OpenOffice jars and native libraries. This allows Docmosis to launch the Converters itself rather ("embedded converters") rather than this requiring a separate script (eg runConverter.sh). This means setup is simpler for smaller systems.</p> <p>The launching works from within more Web Application Servers than before, such as JBoss5, Glassfish etc. Note: using "embedded converters" implies you are running everything on the same machine as opposed to a load-distribution configuration.</p> <p>To use embedded converters, update see the information in the example <i>converterPoolConfig.xml</i> file.</p> <p>The primary new property allowing control over the library loading is:</p> <p><i>docmosis.openoffice.useCustomLoader=true false</i></p> <p>This property defaults to <i>false</i> but if you have linkage errors, particularly when using embedded converters, you can try setting this to <i>true</i> to load the libraries in a different fashion.</p>
3	<p>The property <i>template.store.location</i> can now be left blank, in which case Docmosis will create a temporary area to work with for the template cache.</p>

[Older] Docmosis v2.1.0 Release Notes

November 2009

New Features

#	Change
1	<p>Hyperlink Insertion</p> <p>Docmosis can now insert active hyperlinks into documents. A link placeholder is inserted into the template using a name with a "link_" prefix. For example a field <code>«link_myWebSpace»</code> will look up the data provider using the key "myWebSpace" and render the result as a hyperlink. Further, the data provided can be delimited using a pipe symbol (" ") to name the link differently from the address.</p> <p>For example, given the field above, a data value of "http://www.mywebpace.com/bluk" would be rendered as a hyperlink and display the text "http://www.mywebpace.com/bluk" in the document. If the data value was "mywebpace http://www.mywebpace.com/bluk", the link would be rendered into the document displaying the text "mywebpace".</p> <p>See <i>example1</i> in the download bundle to see it in action.</p>

API Changes

The following API changes should be noted

Class / Interface	Change
DataProviderBuilder	addFile() methods now allow character encoding to be specified.
StoreHelper	New storeTemplate() method that can take an InputStream as the source of the template rather than just file-based templates.
DropStoreHelper	New process methods to allow Zip and Jar files of templates to be processed directly, or from URLs to Resources from a class loader.

Bug Fixes / Technical Enhancements

#	Change
1	<p>Expressions Enhanced - general improvements</p> <ol style="list-style-type: none">i. default boolean true test <code>cs_{isFriend()}</code> was previously invalid, but now evaluates isFriend() as a boolean as expected.ii. null tests supported <code>cs_{getAlpha()=null}</code>iii. size() capability broadened now also applies to any non-reflective data sources

	iv. isEmpty() capability added cs_{getFriends().isEmpty()}}
2	Fixed issue where rs_, cs_ and es_ tags at the first line of a template page could result in extra blank lines in output documents.
3	Added toString() to TemplateAnalysis implementation to allow dumping of analysis information.
4	Fixed issue with nested anonymous er_ tags failing template registration.
5	Fixed issue with second and subsequent Tables Of Content/Tables of Figures etc not being updated correctly.
6	Improved handling of java.sql.Time data type.
7	Subtle fixes to template caching and warnings for large templates
8	Fixes to DataProviderBuilder.addObject(name, Object) so that it works with Collection/Array data types and allows them to be referenced directly by the name given.
9	The connection to OpenOffice will complain with a specific message about unsupported Java version if the OpenOffice API is not compatible with the version of Java in use.
10	Improved population of non-ascii characters to support multiple languages and symbology.