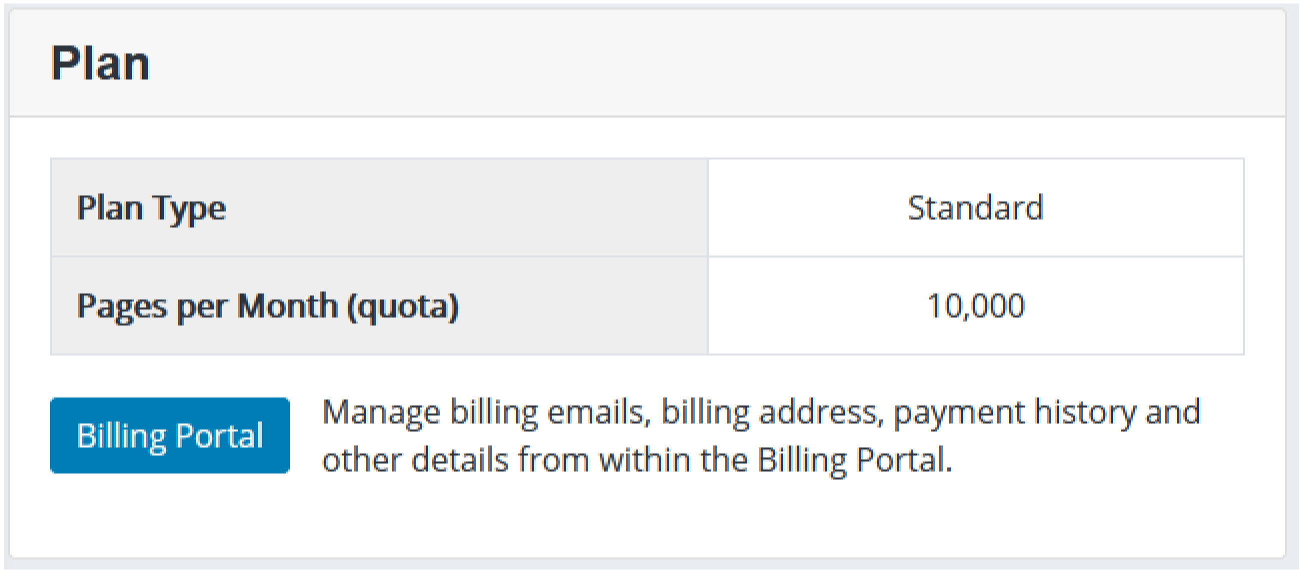
Docmosis Cloud DWS2 Release Notes

July 2019

Cloud Service Changes / API Changes

#	**Change**
1	User Console - Billing and subscription management now available in the Account page:  Plan | Plan Type | Standard | | Pages per Month (quota) | 10,000 | Manage billing emails, billing address, payment history and other details from within the Billing Portal.
2	User Console – improvements to copy a template in-situ using the common “Copy Of” style of name.

New Core Engine Features

#	**Change**
1	

Bug Fixes / Technical Changes

#	Change
1	Updated to latest AWS SDK for interaction with Amazon Web Services
2	User Console internal libraries updated
3	Fix to template copy function which could put the template into a duplicated path instead of the specified folder
4	Fixed issue where rotation of user access key multiple times could cause console logout

Docmosis Cloud DWS2 Release Notes

July 2019

Cloud Service Changes / API Changes

#	**Change**
1	

New Core Engine Features

#	**Change**
1	

Bug Fixes / Technical Changes

#	Change
1	Fixed regression with expression processing causing some Maths expressions to fail to parse.

Docmosis Cloud DWS2 Release Notes

May 2019

Cloud Service Changes / API Changes

#	Change
1	`/getTemplate` end-point can now take multiple `“templateName”` parameters to download multiple templates in one call. Up to 100 can be specified and the result is a zip file.
2	`/getImage` end-point can now take multiple `“imageName”` parameters to download multiple images in one call. Up to 100 can be specified and the result is a zip file.

3	`/uploadImage` end-point can now returns details of the uploaded image in the response. For example:
	
	`{`
	`  "succeeded":true,`
	`  "shortMsg":"img1.png stored.",`
	`  "imageDetails":{`
	`    "name":"img1.png",`
	`    "lastModifiedMillisSinceEpoch":"1556850764000",`
	`    "lastModifiedISO8601":"2019-05-03T10:32:44+0800",`
	`    "sizeBytes":"10923",`
	`    "isSystemTemplate":"false",`
	`    "md5":"d2a2755fce47b60e685d58a852f7708c"`
	`  }`
	`}`

4	New template functions: replaceStr(source, searchFor, replaceWith [, ignoreCase]) – replace all strings inside another replaceFirst(source, searchFor, replaceWith [, ignoreCase]) – replace first string inside another

New Core Engine Features

#	Change
1	

Bug Fixes / Technical Changes

#	Change
1	Fixed corner case issue where an ODT template, with an image bookmarked with a parameterized method that looks up the image would make a bad call for an image on the DataProvider.
2	Fixed issue with deleteTemplate where specifying no templates would result in a non-meaningful error.
3	Fixed bug when using the copy service to copy a template over itself - it would do the wrong thing and create a bad path.
4	Updated aws sdk to latest version for interfacing with amazon

#	Change
5	updated template and image renaming/moving/copying to also clear the destinations from the local cache. This fixes a bug where old destinations would remain in the cache if they existed pre-rename/move/copy
6	Updated error diagnostics to report a permission error explicitly rather than a general error when trying to write to S3 buckets where permission is not granted.
7	Multiple security-related improvements applied to application and infrastructure.

Docmosis Cloud DWS2 Release Notes

Jan 2019

Cloud Service Changes / API Changes

#	**Change**
1	`/deleteTemplate` end-point can now take multiple `"templateName"` parameters to allow it to delete multiple templates in one call.
2	`/deleteImage` end-point can now take multiple `"imageName"` parameters to allow it to delete multiple templates in one call.

New Core Engine Features

#	**Change**
1	

Bug Fixes / Technical Changes

#	Change
1	Fixed issue where a table with the only Docmosis tags being conditional sections or repeat sections could result in table style being lost (including borders).

Docmosis Cloud DWS2 Release Notes

Nov 2018

Cloud Service Changes / API Changes

#	Change
1	None

New Core Engine Features

#	Change
1	

Bug Fixes / Technical Changes

#	Change
1	Improved numbered list processing so that numbered items can continue numbering in lists and sub-lists where items are repeated or conditioned out.
2	Improved error message when a plain text field is mixed with a merge field.
3	Fixed issue where some image substitutions could leave “0.00.0” text in document

Docmosis Cloud DWS2 Release Notes

Jul 2018

Cloud Service Changes / API Changes

#	**Change**
1	None

New Core Engine Features

#	**Change**

1	New template features to help with single quotes in templates:
	– The new `squote()` function will replace the double-quote characters with single quote characters in the literal string. For example:
	`squote('This is Amy"s')`
	gives
	`This is Amy's`
	– The new `$quot` template variable can be used to create single-quoted strings. For example:
	`<<{'This is amy' + $quot + 's'}>>`

Bug Fixes / Technical Changes

#	Change
1	Fixes to the processing of template-expressions with functions (eg <<{map(abc,1,2…}>>) to allow data to provide commas and single-quotes without confusing the functions being called.
2	XML and JSON sample data does not generate "true" and "false" sample date keys for conditional sections that are literally <<cs_true>> or <<cs_false>>.
3	Fixed corner case NPE with repeating over ranges of certain types

Cloud Service Changes / API Changes

#	Change
1	`/uploadTemplate` – on success this end-point now additionally returns details of the uploaded template under the key "templateDetails":

`name`	the template file name
`lastModifiedMillisSinceEpoch`	last modified in milliseconds
`lastModifiedISO8601`	last modified yyyy-MM-dd'T'HH:mm:ssZ
`sizeBytes`	the size in bytes
`isSystemTemplate`	whether a system template ("true" or "false")
`templatePlainTextFieldPrefix`	the prefix used when it was uploaded
`templatePlainTextFieldSuffix`	the suffix used when it was uploaded
`templateDevMode`	the dev mode setting used when it was uploaded
`templateHasErrors`	true if the uploaded template has errors
`templateDescription`	the description uploaded with the template if any
`md5`	the md5 hash code for the template

#	Change
2	New web service end-points for image management: - `createImageFolder` - `renameImageFolder` - `deleteImageFolder` - `copyImageFolder` - `copyImage` - `renameImage`
3	New web service end-point to delete the secondary access key: - `deleteSecondaryAccessKey` This means that after access key rotation has been completed, the secondary key can be deleted via the API.
4	Sample data generation from the console and the /getSampleData end-point has been improved (see Bug Fixes / Technical Changes below).

New Core Engine Features

#	Change
1	Field processing of ranges on data (eg <<rs_people[3-5]>>) has been improved to handle repetition of elements.

#	Change
2	New template functions: - toSentence(data) – convert the text to sentences with capital initials. - isBlank(data) – returns true if the parameter value is null or empty - ifBlank(data, replacement) – if the given data is blank use replacement instead
3	New built in template variables: - $num - the index into the current data element (in a repeating section) starting at 1. - $nl – a new line character that can be appended to strings etc - $nowMS – the current UTC time in milliseconds since the epoch - $nowUTC - the current UTC time and date in ISO 8601 format. This can be formatted using the dateFormat() function into the required format. - $itemidx1, $itemidx2 – for the item number in "stepped" repeating sections - $idx1, $idx2, $itemnum1, $itemnum2.. as shorthand for $i1.$idx, $i2.$idx etc.
4	

Bug Fixes / Technical Changes

#	Change
1	Template functions have been improved to handle parameters and return results better: - Null results are not displayed as null - If null data is passed functions to do with formatting the result is now null instead of an error. This avoids having to check the data value before using the function. - The Map() function could return a pair of single quotes '' when they were specified as a mapped result. This is now a blank as intended. -
2	Corrected processing of explicit [F-L] range directives which were not operating on the "L" directive as documented in more complex cases (eg [F-L2]).
3	Fixed issue with processing variables in repeating sections that use "nested" terminology (eg <<rs_a.b.c>>. This showed as a problem with <<pageBreakNotLast>> failing in such a repeating section.
4	Fixed processing error when processing a template with conditional rows inside repeating rows where no rows were in the conditioned section. This means the conditional rows were redundant, however they should not cause an error.
5	Fixes made to writing errors into documents in Dev Mode. Some cases were resulting in the error not being visible in the document or causing the document to fail to process.
6	Fix to errors being written into header / footer in dev mode. Some conditions mean the errors were not visible.
7	Sample Data Generation has been improved: - nested names (eg <<rs_hotel.floor>>) are now automatically expanded into a nested data structure - sample hyperlinks are better and refer to example.com - JSON format is more compact (vertically) - Ranges (eg <<rs_person[2-3]>>) are removed from the names - Step directives (eg <<rs_people:step2>>) are removed from the names - Set and get variables are now ignored

#	Change
8	Attempted injection of HTML into text boxes now raises a meaningful error message.
9	Errors in template functions now report the function name so that the problem can be identified more easily in complex expressions with multiple functions.
10	Template functions now provide more specific errors about the parameters when they are invalid.
11	The toAlpha(), toAlpha2() and toRoman() functions provide constraints on their parameters and with meaningful error messages. This stops the functions being used beyond sensible use-cases.

Docmosis Cloud DWS2 Release Notes

Mar 2018

Cloud Service Changes / API Changes

#	**Change**
1	None.

New Core Engine Features

#	**Change**
	None.

Bug Fixes / Technical Changes

#	Change
1	Improved internal file loading to allow some files to load that were previously rejected (esp those derived from docx templates)
2	Improved processing of large payloads and defence against over-large payloads
3	Improved file/template delete processing performance
4	Many internal updates to support new cloud console

Docmosis Cloud DWS2 Release Notes

Feb 2018

Cloud Service Changes / API Changes

#	**Change**
1	None.

New Core Engine Features

#	**Change**
	None.

Bug Fixes / Technical Changes

#	Change
1	Fixed issue in where a specific template feature in combination with specific data could cause a loop that never terminated. This would stop the current document completing and could affect the system more generally.
2	Fixed a cleanup routine that was not closing the streams correctly.
3	Fixed issue in Mail System which was causing threads to "leak" over time.

Docmosis Cloud DWS2 Release Notes

Nov 2017

Cloud Service Changes / API Changes

#	Change
1	None.

New Core Engine Features

#	Change
	None.

Bug Fixes / Technical Changes

#	Change
	Improved document merging (inserting a document inside another). This improves accuracy when there are many tables and/or page layout changes.
	Fix to template processing to identify fields better, particularly in docx format templates. Some corner cases were identified where fields near the bottom of a template page could be ignored where Docmosis-expressions using ">" (the greater-than operator) where used.
	Fix to template processing where an error would be raised indicating an invalid template where the template is valid. The processing was failing for some nested conditional and repeating template structures.
	Fix to template processing where hidden blank fields would cause an "array" error rather than be identified or ignored.
	Fix to template processing where image bookmarks overlapping repeating section tags would fail but not identify the problem for the user.
	Fix to template processing where conditional sections contained only set-variable fields and all on one line. This combination would result in an error that was not helpful.
	Fix to ensure UTF-8 encoding is of data in cases where large repeating sections of content are processed regardless of platform.
	Fix to close a stream during shutdown that was not being closed.

Cloud Service Changes / API Changes

#	Change
1	Support has been added for broader characters in template names, image names and output file names. The broader character set allows: - Unicode characters including multibyte European characters - Common punctuation including the spaces, commas and the characters: `.+'()~_#&[]=@-` Of particular note is the behaviour of the Docmosis Cloud Console: when uploading templates, the template name is "Normalized" into NFC form (http://unicode.org/reports/tr15/). This has been done to provide consistent behaviour across browsers on different platforms. When rendering templates via the REST API that have been uploaded with the Docmosis Cloud Console, the template name may need to be normalized into NFC form for it to be found for rendering.

2	The "`render`" service now can return the "streamed" document back as base64 encoded data in the response in JSON or XML format. This is controlled by the new parameter `streamResultInResponse`.
	When streaming back documents to the calling client, the document is normally returned directly as a binary stream.
	
	If `streamResultInResponse` is set to "`y`" or "`true`", the response JSON or XML will contain the document in an element named "`resultFile`". If you have a JSON request for example, the response would look like this:
	
	`{`
	`  "succeeded":true,`
	`  "resultFile":"JVBERi0xLjQKJcOkw7zDtsOfCjIgMCBvYmoK...`
	`}`

3	The Template Upload and Image Upload API end-points have a new parameter to control automatic-normalization (http://unicode.org/reports/tr15/) of template and image names. By default, template and image names will not be normalized. If the new parameter: `normalizeTemplateName` is set to "`y`" or "`true`", the name will be NFC normalized. When rendering or otherwise referencing these files, it will be important to apply Normalization consistently, noting that the Docmosis Cloud Console automatically applies NFC normalization when uploading through browsers.
4	The Template Upload API now validates template names against the extended character sets. Previously this validation was not done allowing templates to be uploaded that had to be renamed before they could be rendered.

New Core Engine Features

#	**Change**

Bug Fixes / Technical Changes

#	Change

Cloud Service Changes / API Changes

#	Change
1	New render "tagging" ability allows renders to be tagged for tracking task-specific statistics. By applying a tag to a render, the tag can be later queried (the new "/getRenderTags" service end-point) to get page and document counts based on specific tags. Specifically: a) A new render parameter "tags" allows stats to be collected against specified tags. b) A new service endpoint "/getRenderTags" queries statistics for chosen months. See the Web Services Guide for more information.
2	New service end-point "`/getSampleData`" can be used to request JSON or XML format data specifically generated for a template. See the Web Services Guide for more information.

3	Updated numFormat function and "number" renderer to be able to correctly process numeric data when formatting using locales. The new parameter (`applyLocaleToInput)` allows the input to be treated as purely numeric (even if textual) when formatting:
	
	`  numFormat (value, format [, locale [, applyLocaleToInput]])`
	
	`eg.  <<{numFormat(0.025789, '#.##0,00', 'nl', false)}>>`
	
	This parameter should be set to "`false`" when passing a `value` that should not be interpreted using the locale. This applies to non-formatted numeric values.
	Normally the given locale is used to interpret the input and then again when formatting. Setting `applyLocaleToInput` to false means the locale is only used when formatting the output.
	The change also applies to the number renderer:
	
	`  renderer=number(format [, locale [, applyLocaleToInput]])`
	eg.
	`  <<value{renderer=number('#.##0,00', 'nl', 'false')}>>`
	
	See the Docmosis Template Guide for more information.

New Core Engine Features

#	**Change**
1	Improved image handling in repeating sections to fix issues with recent versions of Libre Office image handling.

Bug Fixes / Technical Changes

#	Change
1	New behind-the-scenes features to support the new (coming) version of the Web Console
2	Fixed "null" error corner-case that could occur processing complex fields inside tables.
3	Fixed failure to cleanup resources in "/`getTemplateStructure`" end-point
4	Fixed boolean-logic expression processing which was failing for more complex expressions involving the ! (not) operator
5	Barcode data length has been limited to 200 characters to avoid accidental creation of barcodes that are not sensible.

Docmosis Cloud DWS2 Release Notes

May 2017

Cloud Service Changes / API Changes

#	Change
1	The `listTemplates` end-point now returns the MD5 hashcode of the listed templates.
2	The `uploadTemplate` now supports `multipart/form-data` transfers in addition to `application/x-www-form-urlencoded` transfers .
3	Dev cloud accounts are now throttled to reduce server impact.
4	Date parsing in the date-renderer and `dateFormat` functions now automatically perform a best effort possible when no "strict mode" parsing applied rather than raising an error. Previous behaviour was to fail if the strict mode could not find a suitable parsing format (which meant "flexible" parsing date formats like "d" which can parse single or double digit dates would always be failed when applied to a double-digit date).
5	Improved load distribution of production and dev users across nodes.

New Core Engine Features

#	Change
1	Added processing of Form fields (in ODT templates) so that PDF forms can created with data pre-filled.
2	Date parsing can default to "strict with fallback" instead of simply strict or not strict.

Bug Fixes / Technical Changes

#	Change
1	Fixed possible concurrent overwrite of MD5 buffer
2	Fixed corner-case null-error in table analysis when table has more cells in some rows and merging affects borders.
3	Fixed bug in HTML injection where inserting null/blank HTML into a html field within a table would leave "repairme" marker.

Docmosis Cloud DWS2 Release Notes

Jan 2017

Cloud Service API Changes

#	Change
1	The `render` endpoint has been updated to support mime type `application/x-www-form-urlencoded.`
2	New end point "renderForm" to allow documents to be rendered directly from web-forms providers (eg via "web hooks"). Specializations exist for working with FormSite, WuFoo and FormStack.
3	Updated `uploadTemplate` to allow the previous template to be kept if uploading a template with errors in "production" mode. This means uploading a template with errors will leave the previous (working) template in place. New parameter is called: `keepPrevOnFail`

New Core Engine Features

#	Change
1	

Bug Fixes / Technical Changes

#	Change
1	Fixed possible concurrent overwrite of MD5 buffer

Cloud Service API Changes

#	Change
1	The `render` endpoint allows `storeto` to now be empty as well as not specified
2	Updated render end point to set header `X-Docmosis-PagesRendered` when successful.
3	Updated `getTemplateStructure` to correctly detect `TemplateNotFoundException` and return as a 400 failure with the correct message
4	Updated `GetUserDetailsService` to return info about the related dev/primary account

New Core Engine Features

#	Change

1	**If / Else / Else-If Support in Conditional Sections**
	Simple "else" looks like this:
	`<<cs_true>>`
	`true`
	`<<else>>`
	`false`
	`<<es_>>`
	
	"else-if" is written like this:
	`<<cs_isPerson>>`
	` I have a person`
	`<<else_isPlant>>`
	` I have a plant`
	`<<else>>`
	` I have something I didn't expect`
	`<<es_isPerson>>`
	
	Conditional Sections Support the new Expression Syntax (described below)
	Eg:
	`<<cs_{val < 10.0}>>`
	`Low value = <<val>>`
	`<<else_{val > 100.0}>>`
	`High value = <<val>>`
	`<<else>>`
	`Nominal value = <<val>>`
	`<<es_>>`

#	Change
2	**New Expression Engine** 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. Expressions in Docmosis templates are still delimited by the braces (“{“ and “}”) characters. Expressions can now include: • Precedence using the brackets “(“ and “)” characters • Mathematical expressions • Mathematical and String functions • Boolean logic
3	**Direct Expression Evaluation** Fields which are expressions can be used to insert data into documents. For example: `<<{1 + 2 + 3}>>` `<<{round(val/100,2)}>>%` `<<{titleCase(firstName + ‘ ‘ + lastName)}>>`
4	**Assignment of Expressions to Variables** Variables can now be assigned the results of expressions: << $m={expr} >> Eg. `<<$m={round(1+ceil(2*3.5))}>>` `round(1+ceil(2*3.5)) = <<$m>>`
5	**Boolean Logic** `<<cs_{val1 || (val2 && val3)}>>` `val1 is true or both val2 and val3 are true` `<<es_>>`
6	**Maths functions** Typical math functions are supported. `Eg. <<{max(4.522, 4.5)}>>` The round() function has been extended to support an optional precision: `<<{round(1.236)}>>` `<<{round(1.236, 2)}>>` The precision also pads: `<<{round(1.2,5)}>>`

#	Change

7	**String Functions**
	charAt, compareTo, compareToIgnoreCase, concat, endsWith, equals, equalsIgnoreCase, indexOf, lastIndexOf, length, replace (character replacement), startsWith, substring, toLowerCase, toUpperCase, Trim, map, titleCase, split,
	eg:
	`<<{equals(‘a’,’b’)}>>`
	`<<{indexOf(‘abc’,’b’)}>>`
	`<<{startsWith(‘this is’, ‘this’)}>>`
	`<<{titleCase(‘joe blogs’)}>>`

8	**Formatting Functions**
	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.
	`numFormat(<value>, <format>[, <locale> ])`
	`<<{numFormat(value1, ‘###,###.00’)}>>`
	
	`dateFormat(<value>[, <output format>[, <input format> ]])`
	`<<{dateFormat(value1, ‘dd/MM/yy’, ‘dd-MMM-yyyy’)}>>`

9	**Operators** The well known operators are supported: `( ) + - * / % + - = == != < <= > >= && || !`
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	Sub-templates can now set template-variables that are visible to the “master” template and subsequently processed templates.

Bug Fixes / Technical Changes

#	Change
1	Fixed possible InputStream leak when communicating with Amazon S3
2	Updated internal messaging to perform faster and to prepare for new Europe end point coming online.
3	Updated analysis to detect when overlapping sections are caused by an over-zealous image bookmark.
4	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 processing more natural.
5	Increased the default window size for xml processing for templates with large/complex

#	Change
	paragraphs particularly in docx format.
6	Fixed issue where hidden ("_GoBack") bookmarks created by new versions of word (Mac) were causing blank paragraphs to be left behind.
7	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
8	Update to error handling when injecting errors into output document. Now much less likely to inject into an invisible location or disallowed location (which corrupts document).
9	Updated processing to allow sub-templates to set variables that are visible to master template.
10	Fixed issue where looping over including sub-templates more than 2 levels deep would incorrectly determine a cycle in template referencing.
11	Updated number renderer to be able to take a second parameter which identifies the locale to use when parsing/formatting the date.
12	Update - updated hyperlink processing to be able to use variables and variables set from expressions.
13	Updated hyperlink processing to default to "link:" instead of "link_". Link_ is still supported.

Docmosis Cloud DWS2 Release Notes

Jun 2015

Cloud Service API Changes

#	Change
1	New endpoint: `convert` for converting documents without requiring data or templates. This endpoint is considered experimental. Contact Docmosis support if you would like access to this service.
2	Rendered file names can now contain the "+" character. This is specified in the `outputName` parameter.
3	Image data can reference images by downloading from URLs. This feature is not available to all users.
4	RenderClient Java Library (SDK) updates: 1. Defaults to use "dws2" end point 2. Added setting of recent render parameters (passwordProtect, pdfArchiveMode, pdfWatermark, pdfTagged 3. Added JavaDoc to detail all settings 4. Added support for XML payloads
5	New endpoint `ping` to allow simple service reachability test.

New Core Engine Features

#	Change
1	JSON data processing has had the following changes: 1. Anonymous arrays (eg {["item","item"…]}) are now supported to allow very simplistic data cases. 2. Numeric (non-String) values are no longer being truncated (eg 100.0 becoming 100) 3. Improved performance (reduced processing and memory usage)

Bug Fixes / Technical Changes

#	Change
1	Fixed a bug where template-errors rendered into documents (devMode) were not highlighting in red to make them obvious in most documents.
2	Fixed issue flagging some templates as invalid where complex conditional sections were used within a single paragraph.
3	Improved performance of template and image upload (particularly in periods of reduced cluster operation).
4	Security updates have been implemented based on recent internal audits.
5	Added support for "dev/test" accounts which will be made available to some customers.

Cloud Service API Changes

#	Change
1	Render endpoint returns page count The `render` end point now returns the page count in the header of the response on a successful result. It is returned in the header because when a “streamed” result is requested the payload returned is the document itself. This allows the calling program to obtain and account itself for the number of pages rendered during document production. The header is `X-Docmosis-PagesRendered`.

Cloud Service API Changes

#	**Change**

1	New endpoint: getTemplateStructure
	This new end point allows the caller to get the structure of an uploaded template. It returns a JSON format description of the template structure. The simplest case will be a list of fields, for example:
	`“templateStructure”:{`
	`“field.0”:”firstName”,`
	`“field.1”:”lastName”,`
	`}`
	The JSON keys use the terms “field”, “repeat”, “condition” and “image” to instruct what type of item it is, and then an index starting at 0.
	Importantly, items are in and order and nested structure matching the template. This means that fields within repeating sections will be depicted within a matching structure. For example:
	`“templateStructure”:{`
	` “field.0”:”firstName”,`
	` “field.1”:”lastName”,`
	` “repeat.0.addresses”:{`
	` “field.3”:”addressLine1”,`
	` “field.4”:”addressLine2”,`
	` }`
	`}`

2	New render setting for endpoint: render The render endpoint now supports a new setting to improve the PDF output for accessibility/low-vision users. The PDF is tagged so that alt-text is also “readable” by assistive technologies as well as the normal document content. The setting is: `pdfTagged` and values are “y”, “true”, “false”, “n” or not set, in which case it defaults to false.
3	The uploadTemplate service now returns an error code of 400 when the uploaded template has errors and the devMode is set to false / n. Previously it was returning a 500 code.

New Core Engine Features

#	Change

1 New HTML-injection feature. The new tag:

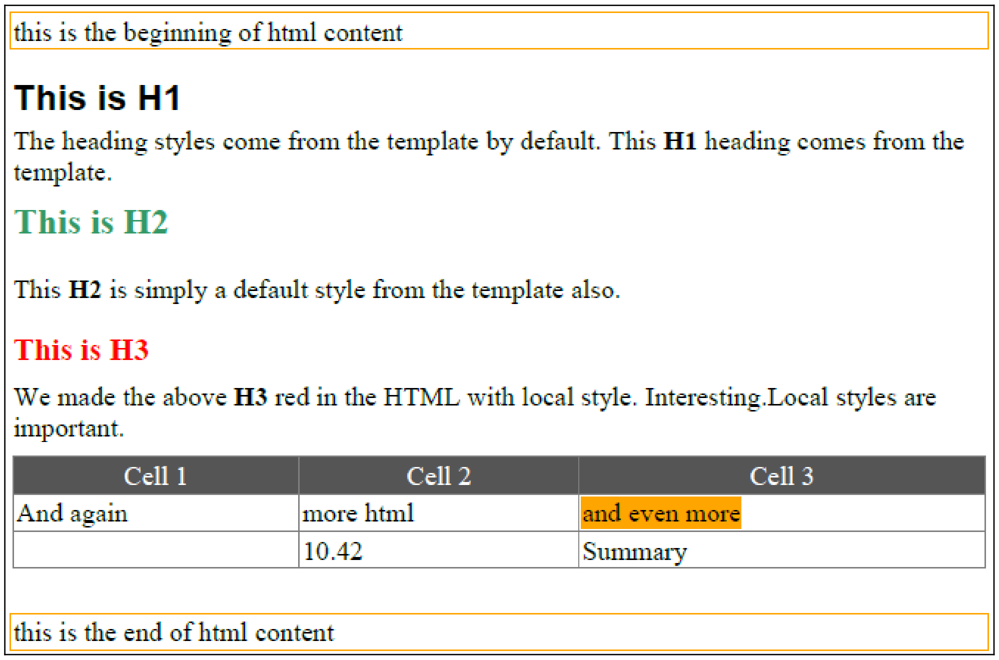
```
<<html:myData>>
```

will process the given data as html and render the html into the document.

So, the following HTML:

```
<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 <b>H1</b> heading comes from the
template.
<h2>This is H2</h2>
This <b>H2</b> 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 <b>H3</b> 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"><span
style="color:white">Cell 1</span></td>
<td style="text-align:center;border:1px solid gray;background-color:#555555"><span
style="color:white">Cell 2</span></td>
<td style="text-align:center;border:1px solid gray;background-color:#555555"><span
style="color:white">Cell 3</span></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"><span style="background-color:orange">and even more</span></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>
```

Will render into a <<html:myData>> field like this:



this is the beginning of html content

This is H1

The heading styles come from the template by default. This **H1** heading comes from the template.

This is H2

This **H2** is simply a default style from the template also.

This is H3

We made the above **H3** red in the HTML with local style. Interesting.Local styles are important.

Cell 1	Cell 2	Cell 3
And again	more html	and even more
	10.42	Summary

this is the end of html content

2 Image Alt Text can now be dynamically changed. Adding a Docmosis tag to the alt-text of an

#	Change
	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: `ConversionInstruction.setPdfTagged(boolean)` 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.
5	Improvements to the Java download: - Library loading and class loading - Confirguration and property setting to allow docmosis embedded converters to be used by multiple applications deployed into same Web or JEE container. Particularly the new Configuration class allows properties to be set/overridden programmatically.

6 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>>”).

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:

a	d	g
b	e	
c	f	

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:

a	e
b	f
c	g
d	

Docmosis will automatically balance the number of rows to fit the given data into the desired number of columns.

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:

#	Change

<<rr_items:step3Down>>		
<<$i1>>	<<$i2>>	<<$i3>>
<<er_>>		

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 name of the person can be referenced:

<<rr_items:step3Down>>		
<<$i1.name>>	<<$i2.name>>	<<$i3.name>>
<<er_>>		

#	Change
7	There is a new tag <<noRowColoring>> (aka <<noRowColouring>>) which disables the automatic row colouring feature of Docmosis when expanding repeating rows in a table. (sometimes table-row colouring is not desirable). The <<noRowColoring>> 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.
8	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. For example, dates: <<myDate{renderer=date('dd/MMM/yyyy')}>> will render myDate into dd/MMM/yyyy And if your date data is in a special format, you can tell Docmosis how to parse it with a second parameter: <<myDate{renderer=date('dd/MMM/yyyy','EEE MMM dd HH:mm:ss zzz yyyy')}>> And booleans: <<myItem{renderer=boolean('yn')}>> will render myItem into y or n values And numbers: <<myVal{renderer=number('$00.00')}>> will render myVal into $00.00 style values. Please see the template guide for more information about the features of the renderers.
9	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. A command-line version is provided as shown by example7 in the Docmosis-Java download. Please see the API for more information.

#	Change
10	Two new built-in variables have been created: $rownum $rowidx 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.

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.
4	Fixed issue where unusual configuration could result in infinite loops in data streaming
5	Fixed issue where occasionally a plain text field might not be recognized as a field in a docX template.
6	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.