Docmosis Cloud v1.10 #3 Release Notes

Nov 2018

Cloud API remains at 1.10.
Docmosis core updated from version 4.1.10 to 4.1.12.

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 v1.10 #2 Release Notes

Jul 2018

Cloud API remains at 1.10.
Docmosis core updated from version 4.1.10 to 4.1.11.

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

Docmosis Cloud v1.10 Release Notes

Jun 2018

Cloud API updated to version 1.10.
Docmosis core updated from version 4.1.9 to 4.1.10.

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.