Release Notes **XMLMill for Java - XMLMill for Domino 2.80**

Table of Contents

1. Preface

The XMLMill application is intended for software developers who want to generate .pdf documents from xml and/or xsl.

For an overview of how to use XMLMill in general please consult in the **`docs/`** directory:.

- **`apidoc/`** -- The JavaDoc concerning the **`XMLMill`** public api.
- **`userguide.pdf`** -- XMLMill user's guide.
- **`dtdguide.pdf`** -- An explanation of XMLMill's elements (tags) and their attributes.
- **`samplesguide.pdf`** -- An overview of the examples in the **`samples/`** directory contained in the download.
- **`digitalsignatures.pdf`** -- An overview of how to digitally sign PDF documents (one pass generation and signing).

If you have questions, please do not hesitate to send a mail to support@xmlmill.com.

☞ *This document is completely generated with XMLMill 2.80 using* **`rnotes280.xml`** *and* **`rnotes.xsl`**. *These files can be found in the directory* **`samples/docs/rnotes`** *in the download.*

2. Upgrading from previous versions

2.1. Upgrading from version 2.70.

No changes on .xsl/.mill/.mill or Java applications are required, but the use of the latest **`xmlmill.xsd`** of this version is required in order to be able to use the new functionalities.

2.2. Upgrading from earlier versions

The best procedure to make your transition as smooth as possible is as follows:

1. Back up all your xml/xsl/mill files or Java code.
2. Read all release notes http://www.xmlmill.com/news.html from your current version to the latest version in correct order (from oldest release notes to newest).
3. Modify your xml/xsl/mill files or Java code accordingly to the release notes.

☞ *Previous versions can be found at http://www.xmlmill.com//download/xmlmillforjava.html#pv or http://www.xmlmill.com//download/xmlmillfordomino.html#pv*

3. What's new ?

3.1. XMLMill for Java

This release notes document describes all new (or enhanced) functionalitites of this release. The main changes are:

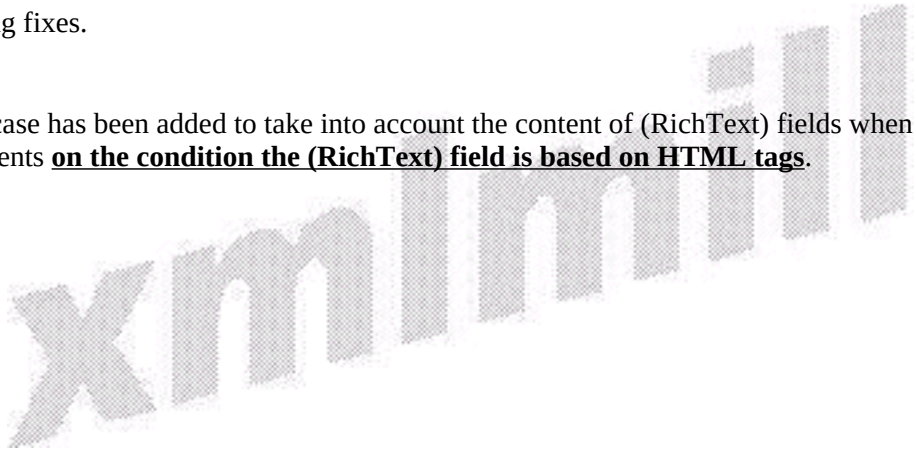
- Availability of the **`space-before`** and **`space-after`** attributes.
- Availability of the **`em`** value for the **`font-size`** attribute.
- Availability of the **`monogram`** value for the **`background-repeat`** attribute.
- Bug fixes.

3.2. XMLMill for Domino

This release notes document describes all new (or enhanced) functionalitites of this release. The main changes are:

- Availability of the **`space-before`** and **`space-after`** attributes.
- Availability of the **`em`** value for the **`font-size`** attribute.
- Availability of the **`monogram`** value for the **`background-repeat`** attribute.
- Convert (RichText) fields with HTML content to PDF.
- Bug fixes.

A new use-case has been added to take into account the content of (RichText) fields when generating PDF documents **on the condition the (RichText) field is based on HTML tags**.



4. XMLMill for Java

4.1. space-before and space-after attributes

As of this version XMLMill supports the **space-before** and **space-after** attributes.

The **space-before** and **space-after** attributes are very similar to the **margin-top** and **margin-bottom** attributes but they are more complicated as explained below.

4.1.1. space-before and space-after

The **space-before** and **space-after** attributes are valid for following elements:

1. **<ml:barcode>**
2. **<ml:leader>**
3. **<ml:p>**
4. **<ml:table>**
5. **<ml:table-and-caption>**
6. **<ml:textbox>**

Unlike the margin properties acting as a single absolute value - meaning they are used to calculate the starting point of an element - the space properties aren't additive.

If there are several space attributes defined between different elements (for example a **space-after** on a preceding element and a **space-before** on a subsequent element), a single space value is chosen, based on the rules for space resolution as defined in the XSL FO specs.

So spaces merge and don't sum up.

This way the space behaviour between elements can be more dynamically managed.

The relation between the relative property and the corresponding absolute property is determined by the "writing-mode" property of the element.

For example, at the top level of a document, if the "writing-mode" is "lr-tb", then **space-before** corresponds to **margin-top**, but if the "writing-mode" is "tb-rl", the "**space-before** corresponds to **margin-right**.

The **space-before** and **space-after** attributes are a shorthand for defining the space-before.minimum, space-before.optimum, space-before.maximum, space-after.minimum, space-after.optimum and space-after.maximum attributes at the same time.

The short form notation can be used together with complete forms; the complete forms have precedence over the expansion of a short form. For example:

```
    space-before="4.0pt"       is equivalent to a specification of
space-before.minimum="4.0pt"    space-before.optimum="4.0pt"
space-before.maximum="4.0pt"    space-before.precedence="0"
space-before.conditionality="discard"
```

4.1.2. space-before.conditionality and space-after.conditionality

Conditionality is an enumerated value which controls whether a space-specifier has effect at the beginning or end of a region. Possible values are **retain** or **discard**; a conditional space-specifier is one for which this value is discard.

4.1.3. space-before.precedence and space-after.precedence

Precedence has a value which is either an integer or the special token **force**. A forcing space-specifier is one for which this value is **force**.

4.1.4. Space-resolution Rules

Currently following rules apply in the current version of XMLMill to determine the distance between elements:

1. If a space-specifiers is conditional and begins a region, it is suppressed, which means that its resolved space-specifier is zero.

2. Further, any conditional space-specifiers which consecutively follow it in the sequence are also suppressed.
3. If any of the remaining space-specifiers is forcing, all non-forcing space-specifiers are suppressed and the value of each of the forcing space-specifiers is taken as its resolved value.
4. Alternatively if all of the remaining space-specifiers are non-forcing, then the resolved space-specifier is defined in terms of those non-suppressed space-specifiers whose precedence is numerically highest, and among these those whose optimum value is the greatest.
5. All other space-specifiers are suppressed. If there is only one of these then its value is taken as its resolved value.

4.2. The `em` value for the `font-size` attribute.

The **`em`** unit is equal to the computed value of the **`font-size`** property of the parent element (following normal CSS rules).

```
 <ml:p font-size="20pt"> Some text <ml:inline font-size="2em"> inline text</ml:inline>
other text</ml:p>
```

The font-size of the inline text will be twice the font-size of its parent element, the paragraph (hence 40pt).

4.3. The `monogram` value for the `background-repeat` attribute.

A monogram is a motif made by overlapping or combining two or more letters or other graphemes to form one symbol. Monograms are often made by combining the initials of an individual or a company, and may be used as recognizable symbols or logos.

The **`background-repeat`** attribute has two new values:

`monogram-horizontal` Shifting the graphic in the next line half horizontally

`monogram-vertical` Shifting the graphic in the next line half vertically.

5. XMLMill for Domino

5.1. space-before and space-after attributes

Please visit the previous section in the previous chapter.

5.2. The em value for the font-size attribute.

Please visit the previous section in the previous chapter.

5.3. The monogram value for the background-repeat attribute.

Please visit the previous section in the previous chapter.

5.4. Convert (RichText) fields with XHTML content to PDF.

As of this version a use-case has been added in the Domino use-case guide explaining how to generate PDF content from data which is stored in (RichText) fields (as XHMLT code).

5.4.1. Way of working

Basically the procedure is as follows:

1. Make an xml-document in a temporary Notes document containing all non RichText fields data (as you did with previous versions of XMLMill).
2. Add a **<richtext>** field in the xml-document containing the RichText field content (so the XHTML tags).
3. Build your xsl-document as in previous versions.
4. Import the **template_rt_html.xsl** stylesheet that contains the templates to convert XHTML tags to **ml:** elements.
5. Apply the **<richtext>** element(s) where needed.

5.4.2. template_rt_html.xsl

The templates are based on the xhtml code generated by the TinyMCE editor (http://tinymce.moxiecode.com), a platform independent web based Javascript HTML WYSIWYG editor control released as Open Source under LPGL by Moxiecode Systems AB.

The advantage of using these templates is that it allows you to fine-tune the PDF output that will be generated (by mofifying the behavior of the templates).

As each xhtml element is converted to one (or more) equivalent ml: element (or elements) you have perfect control over what the output will look like.

In order to be able to convert XHTML elements to **ml:** elements and to succesfully generate PDF documents some conditions must be met:

1. The **<richtext>** contains valid XHTML elements and is a valid XML document.
2. Limit the content of the **<richtext>** element so it only contains the elements defined in this stylesheet or...
3. Extend this stylesheet so it can convert other XHTML elements to the correct **ml:** elements.
4. Be sure to reference in the **config.xml** file all fonts that are referenced in the XHTML element.

Be aware that in the current version of XMLMill:

- Nested tables are not supported.
- Nested list-bullets or list-numbers are not supported.

☞ *This stylesheet does not implement the* **<table>** *element (and descendants) as the TinyMCE*

editor does not define correctly (?) the width of the columns (?). If you know how let us know!

☞ *You can always modify the templates defined in the* **`template_rt_html.xsl`** *stylesheet to suite your needs.*

☞ *If you use another RTEditor and want support from us to modify this stylesheet to match your needs just let us know at: support@xmlmill.com.*

☞ *For more information please consult use-case 5 in the* **`DUsecases.pdf`** *guide contained in the download.*

6. Bugfixes

Following bugfixes were solved:

- In case the attribute **`file`** has not been defined at **`<document>`**level, the name of the generated PDF document is the same as the name of the xml document.
- The base-directory was wrong defined in case the xsl file was in another directory than the xml file (the base directory should always be the directory where the xsl-file is located.
- The base-directory was wrong defined in case the xsl file was defined in the xml-file (the base directory should always be the directory where the xsl-file is located.
- The length and location of a **`<leader>`** element was not always correctly calculated.
- As of this version the **`orphans`** attribute is implemented for the **`<paragraph>`** element.
- The font-size was not correctly calculated in some cases where the **`%`** value was used in an **`ml:inline`** element which was preceded by another element also containg a **`%`** value for the font-size.