

# School math

*Ton Otten*

## Introduction

Since 2004 we are involved in a Dutch OpenSource math project: Math4All. We are responsible for the structure and coding of the educational content. Over the years the content went from TeX coding to HTML/MathML and then to XML/OpenMath, XML/MathML 1.0–3.0 and since 2014 finally XML/AsciiMath.

Math4All tries to advocate itself as a web only curriculum, but its users feel more confident with the support of a paper version. That's where ConT<sub>e</sub>Xt comes in.

Early 2014 the Math4All Foundation took up a combined effort with Malmberg Publishers to publish a complete highschool math curriculum. Math4All is still publishing their content for free while MathPlus takes the commercial route.

Since their cooperation the XML content is used for generating the Math4All website ([www.math4all.nl](http://www.math4all.nl)) and the MathPlus website ([www.mathplus.nl](http://www.mathplus.nl)) and also used in XML2PDF workflows for generating the paper versions.

To give you an idea of the extent of this project: we're talking of 55 text books ( $\pm 220$  pages/book) and answer books ( $\pm 180$  pages/book). So in the end we'll be processing some 22.000 pages.

Below some characteristics of a few chapters:

chapter title	XML files	pages	AsciiMath snippets	images	tables	steptables	exercises
Relations	100	33	603	34	315	0	309
Powerfunctions	141	47	3837	28	5	1425	197

The characteristics and constraints of a math text book are listed below.

text	explaining math takes a lot of text, math and images that are strictly coupled
math	math expressions should never break
images	the staggering number of images and their dimensions (readable) in combination with the no-float constraint
tables	tables have to appear in the text flow and are not allowed to float
exercises	multi-item and single-item exercises that can be open or closed (multiple choice or multiple response) vertical and horizontal alignment of the multi-item denominator and text components (tables, images, formulas)
specific constructs	the stepligntable is a space consuming construct in the text flow
demanding layout	the layout is designed by third parties that have no affinity with math and/or automated typesetting but do want to follow the latest fashions in typesetting
bulky	a large number of XML files, images and generated pages resulting in a considerable runtime

## contextgroup > context meeting 2016

multiple platforms      issues for web and paper have to be dealt with in the XML  
multiple products      proofing versions and final versions of textbook, answerbook and  
work books demand their own 'modes'

We have to deal with the text components and constraints in the XML. That is the reason that we tend to code very redundant. For coding the math schoolbooks we use our own educational DTD that enables us to produce the demanding PDFs and other parties to generate the websites.

In this article we address some of the issues we ran into.

### AsciiMath

At some time the web party in this project decided it would be a good idea to switch from MathML to AsciiMath. So from one day to the other we were confronted with AsciiMath. There was a preliminary AsciiMath handler in ConTeXt at the time, but that was not handling all AsciiMath 'correctly'.

Correctly means here: handling it the way MathJax does. MathJax is an open source javascript that is used to display the math on web pages.

AsciiMath is a simple markup language that looks somewhat like the T<sub>E</sub>X way to code formulas. An expression could look like this:

```
` 3x^2 + 5x - 3 = 17 `
```

The ` (back ticks) denotes the begin and end of the math expression.

In the XML content the expression is coded this way:

```
<am> 3x^2 + 5x - 3 = 17 </am>
```

Which yields:

$$3x^2 + 5x - 3 = 17$$

Below we'll address a few coding issues.

We refer to <http://www.wjagray.co.uk/maths/ASCIIMathTutorial.html> for getting acquainted with AsciiMath.

### Decimal numbers

In Dutch we are used to write a decimal number ( $n \in \mathbb{R}$ ) with a decimal comma instead of a decimal point. Since AsciiMath comes from the USA, the developers didn't bother with other languages. This can lead to strange situations:

```
` sqrt 1,233 `      ` sqrt 1.233 `
```

This becomes:

$$\sqrt{1,233} \quad \sqrt{1,233}$$

AsciiMath doesn't see 1,233 as a number but as a coordinate. So our Dutch authors have to code more extensively:

```
\ sqrt ( 1,233 ) `
```

To get:

$$\sqrt{1,233}$$

One could wonder here why T<sub>E</sub>X and math knowledgeable developers used the parenthesis as a means of grouping.

### Grouping of small and large numbers

In MathPlus large numbers are grouped: 100 000. This is also the case for small numbers: 0,000 001 and numbers that are noted with some accuracy: 100 000,000 001. The grouping of numbers in math and physics is known but in MathPlus the grouping is somewhat aberrant. When the number of digits before or after the decimal comma is smaller than five then there should be no grouping. So don't group 9999 but group 10 000. And also: no grouping in 0,0009 but there is grouping in 0,000 01.

This is set in ConT<sub>E</sub>Xt with:

```
\setupasciimath
  [splitmethod=3,
   symbol={,}]
```

Earlier we stated that AsciiMath interprets a comma in a number differently than we do in the Netherlands, so:

```
` 10000,00001 `
```

Results in:

10 000,00 001

And for AsciiMath that is correct because it 'sees' two separate numbers 10000 and 00001 divided by a comma: a coordinate. You can see the small space after the comma. For a project like MathPlus it would have been better to use the decimal point too as a decimal divider. In that case you would have had the option to let the web or rendering engine take care of the correct language dependent display. Unfortunately they decided against it, so we have to do some last-minute corrections in the resources for the correct grouping.

## contextgroup > context meeting 2016

When we encounter:

```
` 0,00001 `
```

We change that manually into:

```
` 0.00001 `
```

The consequence is that the XML resources for paper and web are not parallel anymore.

### Use of the < operator

AsciiMath accepts the following input:

```
` x < y-4 `
```

Because the content is coded in XML the use of < is not recommended. The use of `lt` is preferred:

```
` x lt y-4 `
```

### AsciiMath logic

The AsciiMath developers state that they code intuitively. Sometimes however that intuition is somewhat strange. Compare for example the next fraction where digits and characters are treated differently:

```
` frac a b ` versus ` frac ab cd `
```

This results in:

$$\frac{a}{b} \text{ versus } \frac{a}{b}cd$$

And when you use digits:

```
` frac 1 2 ` versus ` frac 111 222 `
```

You will get another outcome:

$$\frac{1}{2} \text{ versus } \frac{111}{222}$$

In the first example you have also seen that grouping the characters is necessary to obtain the right fraction  $\frac{ab}{cd}$ :

```
\ frac (ab) (cd) \
```

### Combinations of characters

Certain character combinations have a math meaning in AsciiMath. Known examples are:

```
\ sin alpha \ \ cos beta \ \ tan gamma \
```

This becomes:

$\sin \alpha$   $\cos \beta$   $\tan \gamma$

But there are also character combinations that give unexpected results. Therefore it is advisable to code multi character variables with spaces:

```
\ s p e e d \ versus \ speed \
```

Delivers:

*speed* versus *sped*

In AsciiMath the ee character combination is reserved for Euler's number e.

### Special characters

Some characters have a special meaning.

```
\ a_x n \ versus \ a_g n \
```

Results in:

$a_x n$  versus  $a_g n$

The reason for this interpretation of `_g` is unknown.

### Entities

Entities are not supported in AsciiMath, except for `&lt;`; and `&gt;`; , presumably because the ampersand has a specific function.

## contextgroup > context meeting 2016

### Unicode Math

You can use unicode characters in AsciiMath:

```
` x^2 `      versus ` x² `
` sqrt 3 `   versus ` √ 3 `
` x in RR `  versus ` x ∈ ℝ `
```

This ends up in a font dependent result:

$x^2$  versus  $x^2$   
 $\sqrt{3}$  versus  $\sqrt{3}$   
 $x \in \mathbb{R}$  versus  $x \in \mathbb{R}$

### T<sub>E</sub>X en L<sup>A</sup>T<sub>E</sub>X

The AsciiMath developers are familiar with T<sub>E</sub>X and L<sup>A</sup>T<sub>E</sub>X and so it is no wonder that here and there T<sub>E</sub>X math is being accepted in AsciiMath.

```
` a \cdots b `
```

This becomes:

$a \cdots b$

However, you shouldn't use the T<sub>E</sub>X math commands because just a few are supported.

### Grouping

AsciiMath is very flexible when you code math expressions.

```
` axy ` versus ` axy ` versus ` axxy `
```

The output of this is:

$axy$  versus  $a \times y$  versus  $ax \times y$

However, in some expressions grouping is essential to get the right output or to make the expression more readable in your input file.

Grouping is done with parenthesis ( ) or with the curly brace-colon combination { : : }. Because the parenthesis can have a real meaning in the math expression it is advisable to group with: { : : }.

For example:

```
` ( f(x) ) / ( g(x) ) = 3 `
```

Is less readable then:

$$\backslash \{ : f(x) : \} / \{ : g(x) : \} = 3 \backslash$$

And sometimes grouping is essential to get the right result:

$$\backslash (a/b)/(c/d)=\{ : (ad)/(bd) : \}/\{ : (bc)/(bd) : \}=(ad)/(bc)=(ad)/(bc) \backslash$$

Which results in:

$$\frac{\frac{a}{b}}{\frac{c}{d}} = \frac{\frac{ad}{bd}}{\frac{bc}{bd}} = \frac{ad}{bc} = \frac{ad}{bc}$$

Without the { : } grouping it would have resulted in:

$$\frac{\frac{a}{b}}{\frac{c}{d}} = \frac{ad}{bd} / \frac{bc}{bd} = \frac{ad}{bc} = \frac{ad}{bc}$$

### Negative numbers

In MathPlus a negative number is being coded as follows:

$$\backslash x + \text{text}(-) 3 = 7 \backslash$$

This results in:

$$x + -3 = 7$$

In Math4All they use a 'high minus':

$$\backslash x + ^{-}3 = 7 \backslash$$

This becomes:

$$x + ^{-}3 = 7$$

### AsciiMath versus MathML

Marking down MathML to AsciiMath is convenient for authors but even in schoolmath there are math expressions that can't be done in AsciiMath. That is the reason that we always need the escape to handle the math in MathML.

Sometimes there are educational reasons to display a wrong math expression or to highlight (color, bold) parts of an expression. In order to do that you have to have access to those parts of the expression. This is not possible in AsciiMath.

```

<m:math xmlns:m="http://wherever/mathml">
  <m:mrow>
    <m:mfrac>
      <m:mn>6</m:mn>
      <m:mrow>
        <m:menclose notation="updiagonalstrike">
          <m:mn>38</m:mn>
        </m:menclose>
      </m:mrow>
    </m:mfrac>
    <m:mo>&#x22C5;</m:mo>
    <m:menclose notation="updiagonalstrike">
      <m:mn>38</m:mn>
    </m:menclose>
    <m:mo>&#x22C5;</m:mo>
    <m:mn>19</m:mn>
    <m:mo>=</m:mo>
    <m:mfrac>
      <m:mn>3</m:mn>
      <m:mrow>
        <m:menclose notation="updiagonalstrike">
          <m:mn>19</m:mn>
        </m:menclose>
      </m:mrow>
    </m:mfrac>
    <m:mo>&#x22C5;</m:mo>
    <m:mn>38</m:mn>
    <m:mo>&#x22C5;</m:mo>
    <m:menclose notation="updiagonalstrike">
      <m:mn>19</m:mn>
    </m:menclose>
  </m:mrow>
</m:math>

```

This becomes:

$$\frac{6}{38} \cdot 38 \cdot 19 = \frac{3}{19} \cdot 38 \cdot 19$$

### Schoolmath specific constructs

In school math you tend to teach students how to solve problems and sometimes you want to tell them stepwise what to do and how to think.

ConT<sub>E</sub>Xt comes with two constructs: the stepchart and the stepaligntable. We give an example of both and for further reading you should see the *Steps* manual that is in the distribution.



```

<stepchart>
  <cell> <am>(5^3)/25-5^(6-4) =</am> </cell>
  <text> calculate <am>5^3</am> </text>
  <cell> <am>125/25-5^(6-4) =</am> </cell>
  <text> calculate <am>6-4</am> </text>
  <cell> <am>125/25-5^2 =</am> </cell>
  <text> calculate <am>5^2</am> </text>
  <cell> <am>125/25-25 =</am> </cell>
  <text> divide <am>125 / 25</am> </text>
  <cell> <am>5-25 =</am> </cell>
  <text> subtract </text>
  <cell> <am>text(-)20</am> </cell>
  <cell> </cell>
</stepchart>

```

This becomes:

calculate  $5^3$     calculate  $6 - 4$     calculate  $5^2$     divide  $\frac{125}{25}$     subtract

$$\frac{5^3}{25} - 5^{6-4} = \frac{125}{25} - 5^{6-4} = \frac{125}{25} - 5^2 = \frac{125}{25} - 25 = 5 - 25 = -20$$

The stepaligntable construct is very popular in the MathPlus content and it is used over 10.000 times.

```

<stepaligntable>
  <cells>
    <c1><am>200 sqrt(x+30) - 100</am></c1>
    <c2><am>=</am></c2>
    <c3><am>0</am></c3>
  </cells>
  <text>both sides <am>+ 100</am></text>
  <cells>
    <c1><am>200 sqrt(x+30)</am></c1>
    <c2><am>=</am></c2>
    <c3><am>100</am></c3>
  </cells>
  <text>both sides <am>: 200</am></text>
  <cells>
    <c1><am>sqrt(x+30)</am></c1>
    <c2><am>=</am></c2>
    <c3><am>0,5</am></c3>
  </cells>

```

```

<text>both sides exponent <am>2</am></text>
<cells>
  <c1><am>x+30</am></c1>
  <c2><am>=</am></c2>
  <c3><am>0,25</am></c3>
</cells>
<text>both sides <am>- 30</am></text>
<cells>
  <c1><am>x</am></c1>
  <c2><am>=</am></c2>
  <c3><am>text(-)29,75</am></c3>
</cells>
<text/>
</stepaligntable>

```

This becomes:

$$\begin{array}{l}
 200\sqrt{x+30} - 100 = 0 \\
 200\sqrt{x+30} = 100 \\
 \sqrt{x+30} = 0,5 \\
 x + 30 = 0,25 \\
 x = -29,75
 \end{array}$$

both sides +100  
 both sides : 200  
 both sides exponent 2  
 both sides -30

### Automatic typesetting: yes or no

We developed the XML2PDF workflow that is running on local machines of editors but also on a web server that enables authors to use a PDF Service to proof their content. The publisher provides authors with a webbased authoring tool and the content is saved in an XML repository. The PDF Service gets the XML and other resources from the repository and generates the proofing PDFs.

During the proofing process we tend to say that we are indeed typesetting automatically:

- extensive numbering of exercises, examples and chapters
- referencing to the numbered components
- generating tables of contents for the book and chapteropenings
- generating indexes
- converting AsciiMath
- typesetting tables
- placing images
- etc

Because we're using ConT<sub>e</sub>Xt we can be certain that math is typeset correctly, the tocs and registers are okay, the numbering and referencing is correct and that generally everything looks good.

But in the end phase of book production some issues occur that we have to solve manually. That means we have to check the PDF file visually and manually add directives to the XML that are needed for:

- page breaking  
In most constructs we keep content together at a pagebreak or we do a `\testpage`. But we can't prevent page breaking at unfortunate locations. An XML directive makes it possible to invoke a `\page`.

```
<?context-directive injector page versie-1-1 ?>
```

These kind of injectors have a version field. After an extensive update the directive may not be needed anymore and we can disable it by using a new version value.

- line breaking  
School math is supposed not to hyphenate. That means that longer math snippets end up in the margin or interword spacing becomes unacceptable. In that case we insert an XML directive to invoke a `\crlf`.
- image positioning and dimensions  
The number of images on some pages is staggering. In school math where image and text are intertwined we are not allowed to let the image float to another location.  
Furthermore some images are wide and/or high. The image designers have strict dimensions to take into account while designing their images. But a two-sentence question with a 9 cm × 3 cm image produces a lot of white-space on the page. And paper wasting whitespace is something no publisher wants in his books.  
The default image positions on a page are:

here	the image is placed in the text column
column	the image goes to the flight column
column-w	the image is wider than the text column and is allowed to run into the left margin
column-t	the image is positioned in the left margin and text runs around the image

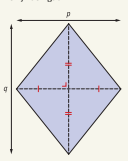
And eventually we can adapt the width of an image to some extent with the constraint that an image must be readable on paper.

26 **DOMEIN** Meten en tekenen

**LITLEG 1 OPGAVE 1** In welke figuren kan een vierhoek altijd worden verdeeld?

**LITLEG 1 OPGAVE 2** Bekijk de figuur in de uitleg. Welke uitspraken zijn waar?  
 A Lijn  $AC$  verdeelt de vierhoek in  $\triangle ABC$  en  $\triangle ABD$ .  
 B Lijn  $BD$  verdeelt de vierhoek in  $\triangle ABD$  en  $\triangle BCD$ .  
 C oppervlakte ( $\triangle ABC + \triangle ACD$ ) = oppervlakte ( $\triangle ABD + \triangle BCD$ )  
 D oppervlakte ( $\triangle BCD$ ) =  $\frac{1}{2} \cdot BD \cdot AC$

**LITLEG 1 OPGAVE 3** Een vierhoek  $ABCD$  wordt gevormd door de punten:  
 $A(1, 2), B(4, -2), C(7, 2)$  en  $D(6, 7)$   
 a Teken vierhoek  $ABCD$ .  
 b Teken diagonaal  $AC$ . Zo ontstaan twee driehoeken.  
 c Teken in beide driehoeken een hoogtelijn.  
 d Bereken de oppervlaktes van de twee driehoeken.  
 e Wat is de oppervlakte van figuur  $ABCD$ ?

**UITLEG 2**  
**ONDERWERP:** hoe bereken je de oppervlakte van een vlieger of een ruit?  
 Bekijk de figuur:  
 In de vlieger zijn diagonalen getekend. De langste diagonaal verdeelt de vlieger in twee even grote driehoeken, die precies dezelfde vorm hebben. De oppervlakte van een vlieger is gelijk aan de oppervlakte van deze twee driehoeken. Door de symmetrie van de figuur hebben deze twee driehoeken allebei basis 30 en hoogte 130.  
 De oppervlakte van de vlieger is dus twee keer de oppervlakte van zo'n driehoek:  
 oppervlakte vlieger =  $2 \cdot \frac{1}{2} \cdot 30 \cdot 130 = 3900$   
 Omdat een vlieger symmetrisch is, geldt dit altijd.  
 De oppervlakte van een vlieger met diagonalen  $p$  en  $q$  bereken je met de formule:  
 oppervlakte vlieger =  $\frac{1}{2} \cdot p \cdot q$   
 Bekijk de figuur:  
  
 Een ruit is een speciaal geval van een vlieger, waarbij de diagonalen elkaar precies halverwege snijden. Net zoals bij iedere vlieger bereken je de oppervlakte van een ruit door deze te verdelen in twee driehoeken.

83 **HOOFDSTUK 4** Verbanden

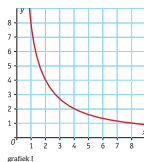
**VOORBEELD 1 OPGAVE 1** Simon heeft 48 chocoladerepen en deelt deze uit. Hoe meer kinderen hij chocoladerepen geeft, hoe minder ieder kind krijgt.  
 a Vul de tabel in met daarin het aantal kinderen  $A$  en de hoeveelheid repen  $A$ . Ga ervan uit dat Simon elk kind steeds evenveel geeft.

$A$ (aantal kinderen)	1	2	3	4	6	8	12	16
$A$ (aantal repen)								


b Geef een formule voor het verband tussen het aantal kinderen  $A$ , onder wie de repen worden verdeeld en het aantal repen  $A$  dat de kinderen krijgen.

**VOORBEELD 1 OPGAVE 2** Een auto met een volle benzinetank kan een afstand  $s$  van 600 kilometer afleggen. De inhoud van de tank  $L$  is 40 liter.  
 a Hoeveel kilometer  $s$  kan de auto afleggen met een half gevulde benzinetank?  
 b Wat is  $s$  voor  $L = 10$ ?  
 c Geef de formule voor het omgekeerd evenredig verband tussen de afstand  $s$  en het aantal liter  $L$ .

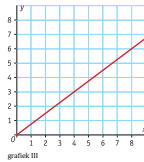
**VERWERKEN**  
 Bekijk de vier figuren en geef aan welke bewering juist is.



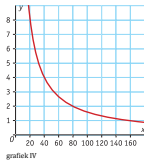
grafiek I



grafiek II



grafiek III



grafiek IV

A Grafiek I is een omgekeerd evenredig verband.  
 B Grafiek II is een recht evenredig verband.  
 C Grafiek III is een omgekeerd evenredig verband.  
 D Grafiek IV is een recht evenredig verband.

## Some afterthoughts

We introduced some nice features for proofing, like 'checking for suspects' (i.e. suspicious spaces, etc.), checking the quality of images, specific checks for AsciiMath and ofcourse the PDF spell checking. Unfortunately these features are seldom used in the PDF Service.

When your working on a project like this in first instance you tend to rely on all the goodies of ConT<sub>E</sub>Xt like the mechanisms for section numbering and referencing, image positioning, relative dimensions and so on. But when you would look at the styles now, you would see a lot of 'hacks' and hard coded stuff. But also remnants of nice solutions that are commented out because the publisher (or someone else) changed his mind somewhere along the line. But then, what else is new.