HANS HAGEN

CONTEXT MKIV

UNITS

Contents

1 Introduction

In ConTEXt MkII there is a module that implements consistent typesetting of units (quantities and dimensions). In MkIV this functionality is now part of the physics core modules. This is also one of the mechanisms that got a new user interface: instead of using commands we now parse text. Thanks to those users who provided input we're more complete now that in MkII. You can browse the mailing list archive to get some sense of history.

2 The main command

The core command is `\unit`. The argument to this command gets parsed and converted into a properly typeset dimension. Normally there will be a quantity in front.

`10 meter`	10 m
`10 meter per second`	10 m/s
`10 square meter per second`	10 m^2/s

The parser knows about special cases, like synonyms:

`10 degree celsius`	10 °F
`10 degrees celsius`	10 °F
`10 celsius`	10 °F

The units can be rather complex, for example:

```
\unit{30 kilo pascal square meter / second kelvin}
```

This comes out as: 30 kPa·m^2/s·K. Depending on the unit at had, recognition is quite flexible. The following variants all work out ok.

`10 kilogram`	10 kg
`10 kilo gram`	10 kg
`10 k gram`	10 kg
`10 kilo g`	10 kg
`10 k g`	10 kg

`10 kg`	10 kg
`10 kilog`	10 kg
`10 kgram`	10 kg

Of course being consistent makes sense, so normally you will use a consistent mix of short or long keywords.

You can provide a qualifier that gets lowered and appended to the preceding unit.

```
\unit{112 decibel (A)}
```

This gives: 112 dB_A. Combinations are also possible:

`5 watt per meter celsius`	5 W/m·°F
`5 watt per meter degrees celsius`	5 W/m·°F
`5 watt per meter kelvin`	5 W/m·K
`5 watt per meter per kelvin`	5 W/m/K
`10 arcminute`	10′
`10 arcminute 20 arcsecond`	10′ 20″

3 Extra units

To some extent units can be tuned. You can for instance influence the spacing between a number and a unit:

```
                        \unit{35 kilogram per cubic meter}
\setupunit[space=normal] \unit{35 kilogram per cubic meter}
\setupunit[space=big]    \unit{35 kilogram per cubic meter}
\setupunit[space=medium] \unit{35 kilogram per cubic meter}
\setupunit[space=small]  \unit{35 kilogram per cubic meter}
\setupunit[space=none]   \unit{35 kilogram per cubic meter}
```

Of course no spacing looks rather bad:

35 kg/m^3
35 kg/m^3
35 kg/m^3
35 kg/m^3
35 kg/m^3
35kg/m^3

Another parameter is `separator`. In order to demonstrate this we define an extra unit command:

```
\defineunit[sunit][separator=small]
\defineunit[nunit][separator=none]
```

We now have two more commands:

```
\unit {35 kilogram cubic meter}
\sunit{35 kilogram cubic meter}
\nunit{35 kilogram cubic meter}
```

These three commands give different results:

35 kg·m^3
35 kg m^3
35 kgm^3

Valid separators are `normal`, `big`, `medium`, `small`, `none`. You can let units stand out by applying color or a specific style.

```
\setupunit[style=\bi,color=maincolor]
\unit{10 square meter per second}
```

Keep in mind that all defined units inherit from their parent definition unless they are set up themselves.

10 m^2/s

To some extent you can control rendering in text and math mode. As an example we define an extra instance.

```
\defineunit[textunit][alternative=text]
```

```
test  \unit    {10 cubic meter per second}  test
test  \textunit{10 cubic meter per second}  test
test $\unit    {10 cubic meter per second}$ test
test $\textunit{10 cubic meter per second}$ test
test  10 \unit    {cubic meter per second}  test
test  10 \textunit{cubic meter per second}  test
test $10 \unit    {cubic meter per second}$ test
test $10 \textunit{cubic meter per second}$ test
```

test 10 m^3/s test
test 10 m^3/s test
test 10 m^3⁄s test
test 10 m^3/s test
test 10 m^3/s test
test 10 m^3/s test
test 10 m^3⁄s test
test 10 m^3/s test

4 Labels

The units, prefixes and operators are typeset using the label mechanism which means that they can be made to adapt to a language and/or adapted. Instead of language specific labels you can also introduce mappings that don't relate to a language at all. As an example we define some bogus mapping.

```
\setupunittext
  [whatever]
  [meter=retem,
   second=dnoces]

\setupprefixtext
  [whatever]
  [kilo=olik]

\setupoperatortext
  [whatever]
  [solidus={ rep }]
```

Such a mapping can be partial and the current language will be the default fallback and itselfs falls back on the English language mapping.

```
\unit{10 km/s}
\unit{10 Kilo Meter/s}
\unit{10 kilo Meter/s}
\unit{10 Kilo m/s}
\unit{10 k Meter/s}
```

When we typeset this we get the normal rendering:

10 km/s
10 km/s
10 km/s
10 km/s
10 km/s

However, when we change the language parameter, we get a different result:

10 olikretem rep dnoces
10 olikretem rep dnoces
10 olikretem rep dnoces
10 olikretem rep dnoces
10 olikretem rep dnoces

The alternative rendering is set up as follows:

```
\setupunit[language=whatever]
```

You can also decide to use a special instance of units:

```
\defineunit[wunit][language=whatever]
```

This will define the `\wunit` command and leave the original `\unit` command untouched.

5 Digits

In addition to units we have digits. These can be used independently but the same functionality is also integrated in the unit commands. The main purpose of this command is formatting in tables, of which we give an example below.

`12,345.67 kilogram`	12,345.67 kg
`__,__1.23 kilogram`	1.23 kg
`__,___.12 kilogram`	.12 kg
`__,__1.== kilogram`	1 kg
`__,___:23 kilogram`	23 kg

The _ character serves as placeholders. There are some assumptions to how numbers are constructed. In principe the input assumes a comma to separate thousands and a period to separate the fraction.

10 km/s 10 km/s 10 km/s 10 km/s 10 km/s

You can swap periods and commas in the output. In fact there are a few methods available. For instance we can separate the thousands with a small space instead of a symbol.

```
\starttabulate[|c|r|r|]
\HL
\NC 0 \NC \setupunit[method=0]\unit{00,000.10 kilogram}
      \NC \setupunit[method=0]\unit{@@,@@0.10 kilogram} \NC \NR
\NC 1 \NC \setupunit[method=1]\unit{00,000.10 kilogram}
      \NC \setupunit[method=1]\unit{@@,@@0.10 kilogram} \NC \NR
\NC 2 \NC \setupunit[method=2]\unit{00,000.10 kilogram}
      \NC \setupunit[method=2]\unit{@@,@@0.10 kilogram} \NC \NR
\NC 3 \NC \setupunit[method=3]\unit{00,000.10 kilogram}
      \NC \setupunit[method=3]\unit{@@,@@0.10 kilogram} \NC \NR
\NC 4 \NC \setupunit[method=4]\unit{00,000.10 kilogram}
      \NC \setupunit[method=4]\unit{@@,@@0.10 kilogram} \NC \NR
\NC 5 \NC \setupunit[method=5]\unit{00,000.10 kilogram}
      \NC \setupunit[method=5]\unit{@@,@@0.10 kilogram} \NC \NR
\NC 6 \NC \setupunit[method=6]\unit{00,000.10 kilogram}
      \NC \setupunit[method=6]\unit{@@,@@0.10 kilogram} \NC \NR
\HL
\stoptabulate
```

0	00,000.10 kg	0.10 kg
1	00.000,10 kg	0,10 kg
2	00,000.10 kg	0.10 kg
3	00 000,10 kg	0,10 kg

4	00 000.10 kg	0.10 kg
5	00 000,10 kg	0,10 kg
6	00 000.10 kg	0.10 kg

The digit modes can be summarized as::

1. periods/comma
2. commas/period
3. thinmuskips/comma
4. thinmuskips/period
5. thickmuskips/comma
6. thickmuskips/period

You can reverse the order of commas and period in the input by setting the parameter `order` to `reverse`.

The digit parser handles a bunch of special characters as well as different formats. We strongly suggest you to use the grouped call.

`.`	, .	comma or period
`,`	, .	comma or period
`:`		invisible period
`;`		invisible comma
`_`		invisible space
`/`		invisible sign
`-`	−	minus sign
`+`	+	plus sign
`//`		invisible high sign
`--`	-	high minus sign
`++`	+	high plus sign
`=`	-	zero padding

Let's give some examples:

`1`	1
`12`	12
`12.34`	12.34
`123,456`	123,456
`123,456.78`	123,456.78
`12,34`	12,34
`.1234`	.1234
`1234`	1234
`123,456.78^9`	$123{,}456.78\times10^{9}$
`123,456.78e9`	$123{,}456.78\times10^{9}$
`/123,456.78e-9`	$123{,}456.78\times10^{-9}$
`-123,456.78e-9`	$-123{,}456.78\times10^{-9}$
`+123,456.78e-9`	$+123{,}456.78\times10^{-9}$

`//123,456.78e-9`	123,456.78×10^{-9}
`--123,456.78e-9`	-123,456.78×10^{-9}
`++123,456.78e-9`	+123,456.78×10^{-9}
`___,___,123,456,789.00`	123,456,789.00
`___,___,_12,345,678.==`	12,345,678

6 Adding units

It is possible to add extra snippets. This is a two step process: first some snippet is defined, next a proper label is set up. In the next example we define a couple of TEX dimensions:

```
\registerunit
  [unit]
  [point=point,
   basepoint=basepoint,
   scaledpoint=scaledpoint,
   didot=didot,
   cicero=cicero]
```

Possible categories are: `prefix`, `unit`, `operator`, `suffix`, `symbol`,`packaged`. Next we define labels:

```
\setupunittext
  [point=pt,
   basepoint=bp,
   scaledpoint=sp,
   didot=dd,
   cicero=cc]
```

Now we can use use these:

```
\unit{10 point / second}
```

Of course you can wonder what this means.

10 pt/s

When no label is defined the long name is used:

```
\registerunit
  [unit]
  [page=page]
```

This is used as:

```
\unit{10 point / page}
```

Which gives:

10 pt/page

7 Built in keywords

A given sequence of keywords is translated in an list of internal keywords. For instance `m`, `Meter` and `meter` all become `meter` and that one is used when resolving a label. In the next tables the right column mentions the internal keyword. The right column shows the Cased variant, but a lowercase one is built-in as well.

The following prefixes are built-in:

Atto	atto
Centi	centi
Deca	deca
Deci	deci
Exa	exa
Exbi	exbi
Femto	femto
Gibi	gibi
Giga	giga
Hecto	hecto
Kibi	kibi
Kilo	kilo
Mebi	mebi
Mega	mega
Micro	micro
Milli	milli
Nano	nano
Pebi	pebi
Peta	peta
Pico	pico
Root	root
Tebi	tebi
Tera	tera
Yobi	yobi
Yocto	yocto
Yotta	yotta
Zebi	zebi
Zepto	zepto
Zetta	zetta

E	exa
G	giga
M	mega
P	peta
T	tera

Y	yotta
Z	zetta
a	atto
c	centi
d	deci
da	deca
f	femto
h	hecto
k	kilo
m	milli
n	nano
p	pico
u	micro
y	yocto
z	zetto

The following units are supported, including some combinations:

AMU	atomicmassunit
Ampere	ampere
Angstrom	angstrom
Astronomical Unit	astronomicalunit
Atm	atmosphere
Atmosphere	atmosphere
Atomic Mass Unit	atomicmassunit
Bar	bar
Barn	barn
Baud	baud
Bel	bel
Bequerel	bequerel
Bit	bit
Byte	byte
Cal	calorie
Calorie	calorie
Candela	candela
Celsius	celsius
Coulomb	coulomb
Dalton	dalton
Day	day
Degree Celsius	celsius
Degree Fahrenheit	fahrenheit
Degrees Celsius	celsius
Degrees Fahrenheit	fahrenheit
Dyne	dyne
Electron Volt	electronvolt
Erg	erg

Erlang	erlang
Fahrenheit	fahrenheit
Farad	farad
Foot	foot
Gal	gal
Gauss	gauss
Gon	gon
Grad	grad
Gram	gram
Gray	gray
Hectare	hectare
Henry	henry
Hertz	hertz
Hg	mercury
Hour	hour
Inch	inch
Joule	joule
Katal	katal
Kelvin	kelvin
Knot	knot
Liter	liter
Litre	liter
Lumen	lumen
Lux	lux
Maxwell	maxwell
Meter	meter
Metre	meter
Metric Ton	tonne
Minute	minute
Mol	mole
Mole	mole
Nautical Mile	nauticalmile
Neper	neper
Newton	newton
Oersted	oersted
Ohm	ohm
Pascal	pascal
Phot	phot
Poise	poise
Radian	radian
Rev	revolution
Revolution	revolution
Second	second
Siemens	siemens
Sievert	sievert

Steradian	steradian
Stilb	stilb
Stokes	stokes
Tesla	tesla
Tonne	tonne
Volt	volt
Watt	watt
Weber	weber
basepoint	basepoint
cicero	cicero
didot	didot
eV	electronvolt
page	page
point	point
scaledpoint	scaledpoint

A	ampere
B	bel
Hz	hertz
W	watt
b	bel
g	gram
h	hour
hz	hertz
l	liter
lx	lux
m	meter
min	minute
n	newton
s	second
t	tonne
v	volt
	celsius
	fahrenheit

The amount of operators is small:

OutOf	outof
Per	per
Solidus	solidus
Times	times

*	times
.	times
/	solidus
:	outof

There is also a small set of (names) suffixes:

Cubic	cubic
ICubic	icubic
ILinear	ilinear
ISquare	isquare
Inverse	inverse
Linear	linear
Square	square

+1	linear
+2	square
+3	cubic
-1	ilinear
-2	isquare
-3	icubic
1	linear
2	square
3	cubic
^+1	linear
^+2	square
^+3	cubic
^-1	ilinear
^-2	isquare
^-3	icubic
^1	linear
^2	square
^3	cubic

Some symbols get a special treatment:

	percent
ArcMinute	arcminute
ArcSecond	arcsecond
Degree	degree
Degrees	degree
Percent	percent
Permille	permille
Promille	permille
°	degree
′	arcminute
″	arcsecond

These are also special:

Micron	micron
mmHg	millimetermercury

8 Colofon

author Hans Hagen, PRAGMA ADE, Hasselt NL
version March 22, 2013
website www.pragma-ade.nl – www.contextgarden.net