convert_units

v. 1.7+

Christian Biscombe

1 May 2018

convert_units performs unit conversions, providing a command-line interface to the Units module that accompanies rxntoarb (lib/units.rb in the rxntoarb root directory). convert_units requires Ruby 1.9.3 or newer.

Contents

1 Invocation and command-line options

convert_units is invoked as follows:

```
convert_units [options_list] <input_string> [output_units].
```

input_string should consist of an optional numerical value (assumed to be 1.0 if not specified) and a list of unit abbreviations, all of which should be separated by spaces (or alternatively by the . or * characters). Any valid SI prefix (see Table 1) may immediately precede the unit name. Note the difference between, e.g., ms-1 (per millisecond) and m s-1 (metre per second). Exponents on units should immediately follow the unit name, optionally preceded by the ^ character. Exponents may be integers, fractions, or decimals. Use of a solidus (/) to indicate reciprocal units is not supported; use negative exponents instead. Accepted unit abbreviations are listed by calling convert_units with the -l flag.

The optional output_units should contain the units into which the input is to be converted, following the same conventions described above. If output_units is present then it must have the same dimensions as input_string. If output_units is omitted (or an empty string) then conversion to SI base units will be performed.

Table 1 SI prefixes.

name	prefix	factor	name	prefix	factor
yotta	Y	10^{24}	deci	d	10^{-1}
zetta	Z	10^{21}	centi	c	10^{-2}
exa	E	10^{18}	milli	m	10^{-3}
peta	P	10^{15}	micro	µ*	10^{-6}
tera	T	10^{12}	nano	n	10^{-9}
giga	G	10^{9}	pico	p	10^{-12}
mega	M	10^{6}	femto	f	10^{-15}
kilo	k	10^{3}	atto	a	10^{-18}
hecto	h	10^{2}	zepto	z	10^{-21}
deka	da	10^{1}	yocto	y	10^{-24}

* `convert_units` allows the ASCII character `u` as well as the Unicode symbols µ (U+00B5) and μ (U+03BC) to represent the micro prefix.

The optional `options_list` may include any of the following:

`-a|--arb` Output in *arb* format, i.e. with units in square brackets before the numerical value. Implies `-d`.

`-d|--double-precision` Output is formatted using the letter `d` for exponents (Fortran double precision copy-and-paste mode). This option is applied automatically if `input_string` contains a double precision numerical value.

`-f|--format <format>` Output is formatted using the specified format string. Any format string recognised by Ruby's `Kernel#sprintf` method is valid.

`-l|--list` List all recognised units and their abbreviations.

`-s|--sig-figs` Output with the same number of significant figures as the input. Overrides `-f`.

`-t|--tdiff` Specifies that input temperatures should be interpreted as temperature differences rather than references to absolute temperatures. See § 2 for examples.

`-v|--version` Print version information.

2 Examples

```
> convert_units ‘nM’
1e-06 mol m-3
> convert_units ‘2.5e-3 V cm-1’
0.25 kg m A-1 s-3
> convert_units ‘11.893 mile h-1’ ‘ft s-1’
17.4431 ft s-1
```

Fractional and decimal exponents:

```
> convert_units ‘8.34e-8 cP Da1/3 K-1 cm2 s-1’ ‘Pa s kDa1/3 K-1 m2 s-1’
8.34e-16 Pa s kDa1/3 K-1 m2 s-1
> convert_units ‘1 M0.5’
31.6228 mol0.5 m-1.5
```

arb *format:*

```
> convert_units -a ‘0.84 pmol cm-2 min-1’
[mol m-2 s-1] 1.4d-10
```

Double precision mode:

```
> convert_units ‘2.5d-3 V cm-1’
0.25d0 kg m A-1 s-3
> convert_units -d ‘2.5e-3 V cm-1’
0.25d0 kg m A-1 s-3
```

Custom output formats:

```
> convert_units -f ‘%.3e’ ‘1478.5 kJ kg-1 K-1’ ‘BTU lb-1 °F-1’
3.531e+02 BTU lb-1 °F-1
> convert_units -f ‘%.3f’ ‘1478.5 kJ kg-1 K-1’ ‘BTU lb-1 °F-1’
353.132 BTU lb-1 °F-1
> convert_units -f ‘%.3g’ ‘1478.5 kJ kg-1 K-1’ ‘BTU lb-1 °F-1’
353 BTU lb-1 °F-1
```

Significant figures mode:

```
> convert_units -s ‘10 atm’ ‘kPa’
1.0e+03 kPa
```

Temperature conversions:

```
> convert_units ‘100 °C’
373.15 K
> convert_units -t ‘100 °C’
100 K
> convert_units -- ‘-40 °C’ ‘°F’
-40 °F
```

3 Copyright and licence

`convert_units` source code and documentation © 2017–2018 Christian Biscombe.

`convert_units` is contributed to *arb* finite volume solver (in which copyright is held by Dalton Harvie) under the same licence terms as that project. At the time of writing, *arb* is released under the terms of the GNU General Public License (version 3) as published by the Free Software Foundation.