Smart Units

Andrew Mathas

Version 1.1

This package implements a \SmartUnit macro for converting between the following metric and Imperial units:

	Metric	⟷	Imperial
distance	kilometers	⟷	miles
length	centimeters	⟷	feet and inches
temperature	Celsius	⟷	Fahrenheit
time	24-hour time	⟷	12-hour time
volume	litres	⟷	USA/UK gallons
weight	kilograms	⟷	pounds

Of course, the underlying unit for length and distance is the same but it makes sense to convert between units of similar "scale". The \SmartUnit macro is designed to print only one unit at a time, where the quantity is given in terms of either metric or Imperial units (there is no error checking, however). Depending on the (global) settings the unit will be printed either as a metric unit, or as an Imperial unit, or both. If the required unit is not given as an argument to \SmartUnit then it is computed. The units are printed using `siunitx`, so the formatting of the units can be controlled using \sisetup together with formatting parameters from `siunitx`. Some aspects of the formatting can be controlled "locally" using \SmartUnit and "globally" using \SmartUnitSettings.

The \SmartUnit macro can print units in the following four different formats:

```
\SmartUnit{metric}          % --> metric
\SmartUnit{imperial}        % --> Imperial
\SmartUnit{metric imperial} % --> metric (Imperial)
\SmartUnit{imperial metric} % --> Imperial (metric)
```

For the impatient, here are some examples:

```
\documentclass{article}
\usepackage{smartunits}
\begin{document}
    \SmartUnitSettings{places=2}       % Some global settings
    \SmartUnit{metric imperial}        % Output format: metric (Imperial)
    \SmartUnit{cm=10}                  %  10 cm (3.94′)
    \SmartUnit{feet=4,inches=3}        %  130.98 cm (4′ 3″)
    \SmartUnit{celsius=20}             %  20 °C (68.00 °F)
    \SmartUnit{metric}                 % Output format: metric
    \SmartUnit{miles=5, places=1}      %  8.0 km
    \SmartUnit{imperial}               % Output format: Imperial
    \SmartUnit{hours=12, minutes=1}    %  12:01 PM
    \SmartUnit{km=5,places=3}          %  3.107 mi
    \SmartUnit{imperial metric}        % Output format: Imperial (metric)
    \SmartUnit{miles=5, km=7}          %  5 mi (7 km)
    \SmartUnit{l=1.0, places=1}        %  0.3 gal (1.0 l)
    \SmartUnit{L=1.0, places=1}        %  0.3 gal (1.0 L)
    \SmartUnit{gal=25.0,uk,places=2}   %  25.00 gal (113.65 l)
    \SmartUnit{gal=25.0,usa,places=2}  %  25.00 gal (94.64 l)
\end{document}
```

The calculations done by \SmartUnit are reasonably accurate but LaTeX is not a calculator so rounding and other errors can and do happen (indeed, see the displayed example below). In addition, even assuming that the calculations done by \SmartUnit are accurate this may not be what you want. For example, if the metric unit is given as 100 km then many people will want the Imperial unit to be written as 60 mi, rather than the more accurate 62.137 119 mi. In fact, \SmartUnit can cater for these considerations because all units are printed using the `siunitx` package, which is really clever. The precision of the printed units can be controlled using \sisetup, from the `siunitx` package or by using the short-hands, figures and places, provided by \SmartUnit:

```
\SmartUnit{km=100.0, figures=1} % 60 mi (100 km)
\SmartUnit{km=100.0, places=5}  % 62.136 84 mi (100.000 00 km)
```

(Comparing with the paragraph above, the last calculation is only accurate to three decimal places!) The values of the computed units can be overridden, during proof-reading for example, by specifying both the metric and Imperial units:

```
\SmartUnit{km=100,miles=70} % 70 mi (100 km)
```

Of course, if you specify units incorrectly like this then incorrect values will be printed.

Behind the scenes all units are printed using `siunitx`. Therefore, the precision and formatting of the units can be changed using the options to \sisetup described in the `siunitx` manual. In particular, when the rounding capabilities of `siunitx` are enabled any units calculated by \SmartUnit will be rounded. In this case there is a shorthand for overriding the global `siunitx` settings for the rounding precision on an individual "smart unit":

```
% Locally=\sisetup{round-precision=3, round-mode=places}
\SmartUnit{cm=2, places=3} % 0.787′ (2 cm)
```

Section 5.5 of the `siunitx` manual should be consulted for all of the options for controlling "post-processing" of numbers. In particular, units that are given as integers will always be printed as integers. For example, the metric version of \SmartUnit{cm=2, places=3} is printed as 2 cm whereas the Imperial version is 0.787′. If, instead, we use \SmartUnit{cm=2.0, places=3} then the metric unit is printed as 2.000 cm.

Unit conversions

For all of these examples we use the following global settings:

```
\SmartUnitSettings{places=2, metric imperial} % Global settings
```

Of course, as in some of the example below, this can always be overridden locally.

Distance – km, miles, mi

Conversions between kilometers and miles. Miles can be specified using either **miles** or **mi**.

```
\SmartUnit{km=100,miles=60}          % 100 km (60 mi)
\SmartUnit{km=100.0,figures=1}       % 100 km (60 mi)
\SmartUnit{miles=62.15,places=1}     % 100.0 km (62.2 mi)
```

Length – cm, inches, feet

```
\SmartUnit{feet=7, inches=1}         % 218.42 cm (7′ 1″)
\SmartUnit{cm=189, feet=7, inches=1} % 189 cm (7′ 1″)
\SmartUnit{cm=191, places=1}         % 191 cm (6′ 3″)
```

Temperature – celsius, fahrenheit

```
\SmartUnit{celsius=32,places=3}      % 32 °C (89.600 °F)
\SmartUnit{celsius=0}                % 0 °C (32.00 °F)
\SmartUnit{fahrenheit=0}             % −17.78 °C (0 °F)
```

Time – hours, minutes, seconds, am, pm

```
\SmartUnit{hours=0, minutes=59}          % 0:59 (12:59 AM)
\SmartUnit{hours=12, minutes=12}         % 12:12 (12:12 PM)
\SmartUnit{hours=13, minutes=9}          % 13:09 (1.00:09 PM)
\SmartUnit{hours=8, minutes=31, pm}      % 20.00:31 (8:31 PM)
\SmartUnit{hours=9, minutes=3, am}       % 9:03 (9:03 AM)
\SmartUnit{hours=0, minutes=0,seconds=1} % 0:00:01 (12:00:01 AM)
\SmartUnit{hours=12, minutes=0,seconds=1} % 12:00:01 (12:00:01 PM)
```

Volume – L, l, gal, gallons

Converting between litres and gallons. Liters can be specified using **L** or **l** and gallons can be given with **gallons** or **gal**. Both UK gallons and USA gallons are supported, with USA gallons being the default.

```
\SmartUnit{l=10.0, places=1}         % 10.0 l (2.6 gal)
\SmartUnit{L=10.0, places=1,uk}      % 10.0 L (2.2 gal)
\SmartUnit{gal=10.0, places=2}       % 37.85 l (10.00 gal)
```

Weight – kg, pounds, lbs

```
\SmartUnit{kg=10.0, places=1}        % 10.0 kg (22.0 lbs)
\SmartUnit{pound=10.0, places=1}     % 4.5 kg (10.0 lbs)
\SmartUnit{pound=10.0,figures=1}     % 5 kg (10 lbs)
```

Formatting options

The following options can either be used inside \SmartUnit to control the formatting *locally* for just the unit being printed. They can also be used as arguments to \SmartUnitSettings to change the formatting of all subsequent smart units (unless these settings are overridden locally). More control over the formatting of the units is given by the \sisetup macro from the `siunitx` package.

figures

This is equivalent to \sisetup{round-mode=figures, precision=#1}. All units are printed with #1 significant figures.

```
\SmartUnit{miles=60, metric, figures=0} % 0 km
\SmartUnit{miles=60, metric, figures=1} % 100 km
\SmartUnit{miles=60, metric, figures=2} % 97 km
\SmartUnit{miles=60, metric, figures=3} % 96.6 km
\SmartUnit{miles=60, metric, figures=4} % 96.56 km
```

places

This is equivalent to \sisetup{round-mode=figures, precision=#1}. All units are printed with #1 significant figures.

```
\SmartUnit{miles=60, metric, places=0} % 97 km
\SmartUnit{miles=60, metric, places=1} % 96.6 km
\SmartUnit{miles=60, metric, places=2} % 96.56 km
\SmartUnit{miles=60, metric, places=3} % 96.561 km
\SmartUnit{miles=60, metric, places=4} % 96.5607 km
```

imperial

Print only Imperial units.

```
\SmartUnit{km=1.0, imperial}     % 0.62 mi
\SmartUnit{miles=1.0, imperial} % 1.0 mi
```

imperial metric

Print Imperial units with metric units enclosed in brackets.

```
\SmartUnit{km=1.0, imperial metric}     % 0.62 mi (1.0 km)
\SmartUnit{miles=1.0, imperial metric} % 1.0 mi (1.6 km)
```

metric (default)

Print only metric units.

```
\SmartUnit{km=1.0, metric}     % 1.0 km
\SmartUnit{miles=1.0, metric} % 1.6 km
```

metric imperial

Print metric units with Imperial units enclosed in brackets.

```
\SmartUnit{km=1.0, metric imperial}     % 1.0 km (0.62 mi)
\SmartUnit{miles=1.0, metric imperial} % 1.6 km (1.0 mi)
```

uk

Selects the UK variants of the Imperial units (currently this only affects gallons):

```
\SmartUnit{l=1.0, imperial metric, figures=2,uk}    % 0.22 gal (1.0 l)
\SmartUnit{gal=1.0, imperial metric, figures=2,uk} % 1.0 gal (4.5 l)
```

usa (default)

Selects the USA variants of the Imperial units (currently this only affects gallons):

```
\SmartUnit{l=1.0, imperial metric, figures=2,usa}    % 0.26 gal (1.0 l)
\SmartUnit{gal=1.0, imperial metric, figures=2,usa} % 1.0 gal (3.8 l)
```

By default \SmartUnit uses the American Imperial units. This is a purely democratic choice dictated by the populations of the two countries.

The code

Behind the scenes, the \SmartUnit macro uses `pgfkeys` and `pgfmath` to convert between the different units. The interface is surprisingly simple and easy to extend. For example, the following code takes care of the conversion between kilometers and miles:

```
%%%% convert=distance:  km <-> miles %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
km/.style={convert=distance, unit km=#1},%
mi/.style={convert=distance, unit miles=#1},%
miles/.style={convert=distance, unit miles=#1},%
distance/metric/.style={/SmartUnit/conversion={km}{miles}{0}{1.609344}{0}{km}},%
distance/imperial/.style={/SmartUnit/conversion={miles}{km}{0}{0.62137119}{0}{\text{mi}}},%
```

Most of the conversions are done in this way — the exceptions are "length" and "time", which are more complicated. The key /SmartUnit/conversion is a macro that takes six arguments and then implements the generic conversion formula:

$$\text{New unit}(\#1) = \Big(\text{Old unit}(\#2) + \text{Pre-offset}(\#3)\Big) * \text{Multiplier}(\#4) + \text{Post-offset}(\#5),$$

where #6 is the metric or Imperial symbol for the unit being printed.

Adding additional conversions is quite straightforward and I am happy to incorporate suggestions.

There is, of course, some trickery driving how the units are printed. Here is the definition of \SmartUnit:

```
\newcommand\SmartUnit[1]{%
    \bgroup%=> changes are local to \SmartUnit => no need to reset
        \pgfkeys{/SmartUnit,#1}% Pass the keys to /SmartUnit
        \pgfkeys{/SmartUnit,printunit}% Calculate units and print
    \egroup%
}
```

So \SmartUnit calls `pgfkeys` twice: the first time to set the keys and the second time to print them using /SmartUnit/printunit. In turn, this uses /SmartUnit/convert to identify the unit being printed, after which the pgf key /SmartUnit/output is called. This has subkeys for the four different *output formats*:

`metric`, `imperial`, `metric imperial` and `imperial metric`.

Finally, /SmartUnit/output/<format> uses /SmartUnit/convert to perform the required conversion. This multi-step printing process is necessary for converting between inputs like time or centimeters, feet and inches, where the output units depend on more than one input value. It is also used to print the four different output formats discussed above.

Acknowledgement

The package was written in response to a TeX.SX question, partly as a proof-of-concept and partly as an exercise to learn how to use `pgfkeys`. I thank Mark Adams for asking the question.

Author

Andrew Mathas, © February 2016.

Licence