The lengthconvert package

Marco Daniel E-mail: marco.daniel@mada-nada.de

Released 2013/06/13

Sometimes it's useful for any explanation to provide length in standardizations units. This package can do this for your.

The command is needed for one of my other packages so I decided to write a small package.

Contents

1 Basics

The package needs the newest version of l3kernel available at CTAN. Internally it uses the modul l3fp to convert the length.

All allowed length are listed in the table below.

Table 1: Allow TEX units

Unit	Measurement
pt	point
pc	pica(1pc=12pt)
in	inch (1 in = 72.27 pt)
bp	bigpoint(72bp=1in)
cm	centimeter (2.54 cm = 1 in)
mm	millimeter (10 mm = 1 cm)
dd	didot point (1157 dd = 1238 pt)
cc	cicero (1 cc = 12 dd)
sp	scaled point (65536 sp = 1 pt)

2 Usage

The usage is really simple. At the length to the main command and get the result.

`\Convert` `\Convert[⟨options⟩] {⟨length⟩}`

The command converts the given length to the unit specified by an option. After the conversion the result will be printed.

`\Convertsetup` `\Convertsetup {⟨options⟩}`

Allows the specification of options.

3 Options

The package is simple and the options too.

`unit` The option accept only the abbrevation unit. Allowed units are described in the table above. You can also use only the abbrevation or complete word phrases without the option. The following table list all allowed inputs.

pt	pc	in	bp	cm	mm
dd	cc	sp	point	pica	inch
big-point	centimeter	millimeter	didot-point	cicero	scaled-point

`use-siunitx` It's a bool flag which can be either `true` or `false`. If it is true, the output of the new length is done by the package siunitx using the command `\SI`.

`precision` This option accepts an integer and specifies the precision of the output.

`number-only` It's a bool flag which can be either `true` or `false`. If it's true, only the number is printed.

4 Examples

Some examples are shown in the following table. In the left column you see the input and in the right the output.

`\Convert{36pt}`	1.26526 cm
`\Convert[precision=2]{36pt}`	1.27 cm
`\Convert[use-siunitx]{36pt}`	1.265 26 cm
`\Convert[unit=pt]{2cm}`	56.9055 pt
`\Convert[unit=dd,number-only]{2cm}`	53.18229
`\Convert[pt]{2cm}`	56.9055 pt
`\Convert[scaled-point]{2cm}`	3729359 sp

5 lengthconvert Implementation

```
1 ⟨*package⟩
2 ⟨@@=lconv⟩
3 \ProvidesExplPackage
4   {lengthtconvert}{2013/05/13}{1.0}{Convert length to another unit}
```

Make sure that the version of l3kernel in use is sufficiently new. This will also trap any problems with l3packages (as the two are now tied together, version-wise).

```
5 \@ifpackagelater { expl3 } { 2012/11/21 }
6   { }
7   {
8     \PackageError { lengthtconvert } { Support~package~expl3~too~old }
9       {
10        You~need~to~update~your~installation~of~the~bundles~'l3kernel'~and~
11        'l3packages'.\MessageBreak
12        Loading~lengthtconvert~will~abort!
13      }
14    \tex_endinput:D
15  }
```

Now load the support packages.

```
16 \RequirePackage{ l3keys2e }
```

__lconv_allowed_shortunits_clist
__lconv_allowed_longunits_clist
__lconv_allowed_allunits_clist

Save all allowed units in a clist

```
17 \clist_new:N \l___lconv_allowed_shortunits_clist
18 \clist_set:Nn \l___lconv_allowed_shortunits_clist
19    { pt , pc , in , bp , cm , mm , dd , cc ,sp }
20 \clist_new:N \l___lconv_allowed_longunits_clist
21 \clist_set:Nn \l___lconv_allowed_longunits_clist
22    { point , pica , inch , big-point , centimeter , millimeter ,
23      didot-point , cicero , scaled-point }
24 \clist_new:N \__lconv_allowed_allunits_clist
25 \clist_set:NV \__lconv_allowed_allunits_clist \l___lconv_allowed_shortunits_clist
26 \clist_put_right:NV \__lconv_allowed_allunits_clist \l___lconv_allowed_longunits_clist
```

(*End definition for* `\__lconv_allowed_shortunits_clist`. *This function is documented on page* **??**.)

`\__lconv_unit_tl` Save the default unit in a token list variable and provide them as option

```
\tl_new:N \l___lconv_unit_tl
\keys_define:nn { lengthconvert }
 {
   unit  .tl_set:N = \l___lconv_unit_tl
 }
\keys_set:nn { lengthconvert }
 {
  unit  = cm ,
 }
```

Provide also abbreviation and word of units

```
\tl_new:N \l__lconv_default_unit_tl
\keys_define:nn { lengthconvert }
 {
    pt    .meta:n  =
             {  unit   =   pt     },
    pc    .meta:n  =
             {  unit   =   pc     },
    in    .meta:n  =
             {  unit   =   in     },
    bp    .meta:n  =
             {  unit   =   bp     },
    cm    .meta:n  =
             {  unit   =   cm     },
    mm    .meta:n  =
             {  unit   =   mm     },
    dd    .meta:n  =
             {  unit   =   dd     },
    cc    .meta:n  =
             {  unit   =   cc     },
    sp    .meta:n  =
             {  unit   =   sp     },
    point .meta:n  =
             {  unit   =   pt     },
    pica  .meta:n  =
             {  unit   =   pc     },
    inch  .meta:n  =
             {  unit   =   in     },
```

```
    big-point  .meta:n  =
            {  unit   =   bp     },
    centimeter .meta:n  =
            {  unit   =   cm     },
    millimeter .meta:n  =
            {  unit   =   mm     },
    didot-point .meta:n  =
            {  unit   =   dd     },
    cicero .meta:n  =
            {  unit   =   cc     },
    scaled-point .meta:n  =
            {  unit   =   sp     },
 }
```

(*End definition for* `\__lconv_unit_tl`. *This function is documented on page* **??**.)

`\l___lconv_use_siunitx_bool` Output should be done by sinutix.

```
\keys_define:nn { lengthconvert } {
  use-siunitx   .bool_set:N = \l___lconv_use_siunitx_bool
}
```

(*End definition for* `\l___lconv_use_siunitx_bool`. *This function is documented on page* **??**.)

`\l__lconv_precision_tl` Specify the precision

```
\keys_define:nn { lengthconvert } {
  precision   .tl_set:N = \l___lconv_precision_tl
}
\keys_set:nn { lengthconvert }
 {
  precision  = 5 ,
 }
```

(*End definition for* `\l__lconv_precision_tl`. *This function is documented on page* **??**.)

`\l___lconv_only_num_bool` Only the number should be used

```
\keys_define:nn { lengthconvert } {
  number-only  .bool_set:N = \l___lconv_only_num_bool
}
```

(*End definition for* `\l___lconv_only_num_bool`. *This function is documented on page* **??**.)

Unknown options should be raised an error

```
\keys_define:nn { lengthconvert } {
  unknown .code:n =
    {
      \msg_error:nnx { lengthconvert } { option-unknown }
            { \exp_not:V \l_keys_key_tl }
    }
}
```

`\Convertsetup` User settings

```
\NewDocumentCommand \Convertsetup { m }
 {
  \keys_set:nn { lengthconvert } { #1 }
 }
```

(*End definition for* `\Convertsetup`. *This function is documented on page* 2.)

`\Convert` Expandable definition of the main command

```
\DeclareExpandableDocumentCommand \Convert { O{} m }
 {
  \group_begin:
    \keys_set:nn { lengthconvert } { #1 }
    \clist_if_in:NVTF \__lconv_allowed_allunits_clist \l___lconv_unit_tl
     {
      \bool_if:NTF \l___lconv_use_siunitx_bool
       {
        \__lconv_using_siunitx:n { #2 }
       }
       {
        \__lconv_nousing_siunitx:n { #2 }
       }
     }
     {
      \msg_error:nnx { lengthconvert } { unit-unknown }
            { \exp_not:V \l___lconv_unit_tl }
     }
  \group_end:
```

```
}
```

(*End definition for* `\Convert`. *This function is documented on page* 2.)

`\__lconv_using_siunitx` Output using siunitx

```
\cs_new:Npn \__lconv_using_siunitx:n #1
 {
  \bool_if:NTF \l___lconv_only_num_bool
   {
    \num
     {
       \fp_eval:n
         {
          round( \dim_to_fp:n { #1 } / 1\l___lconv_unit_tl , \l___lconv_precision_tl)
         }
     }
   }
   {
    \SI
     {
       \fp_eval:n
         {
          round( \dim_to_fp:n { #1 } / 1\l___lconv_unit_tl , \l___lconv_precision_tl)
         }
     }
     {
      \l___lconv_unit_tl
     }
  }
 }
```

(*End definition for* `\__lconv_using_siunitx`. *This function is documented on page* **??**.)

`\__lconv_nousing_siunitx` Output using siunitx

```
\cs_new:Npn \__lconv_nousing_siunitx:n #1
 {
  \bool_if:NTF \l___lconv_only_num_bool
   {
    \fp_eval:n
      {
```

```
        round( \dim_to_fp:n { #1 } / 1\l___lconv_unit_tl , \l___lconv_precision_tl)
       }
    }
    {
     \fp_eval:n
       {
        round( \dim_to_fp:n { #1 } / 1\l___lconv_unit_tl , \l___lconv_precision_tl)
       } \,
     \l___lconv_unit_tl
    }
 }
```

(*End definition for* `\__lconv_nousing_siunitx`. *This function is documented on page* ??.)

```
\msg_new:nnnn { lengthconvert } { option-unknown }
  { Unknown~option~'#1'~for~package~#2. }
  {
    LaTeX~has~been~asked~to~set~an~option~called~'#1'~
    but~the~#2~package~has~not~created~an~option~with~this~name.
  }

\msg_new:nnnn { lengthconvert } { unit-unknown }
  { Unknown~unit~'#1'~for~package~#2. }
  {
    You~are~setting~an~unit~'#1'~which~
    is~unknonw~for~the~package~#2.
  }
```

Finally apply the settings given at load time.

```
\ProcessKeysOptions { lengthconvert }
```

⟨/package⟩

Change History

v1.0

General: First official release 1

Index

The italic numbers denote the pages where the corresponding entry is described, numbers underlined point to the definition, all others indicate the places where it is used.