The **calcage** package*

Robin Schneider

ypid23@aol.de

September 9, 2012

Abstract

The calcage package can calculate the age in years.
Location on CTAN: http://www.ctan.org/pkg/calcage
Fork me on GitHub: https://github.com/ypid/latex-packages/tree/master/calcage

Contents

1 Introduction

The calcage package can calculate the age of someone or something in years. Internally it uses the datenumber package to calculate the age in days. The conversion from days to years is implemented by this package. It is also taken care about leap years and such odd things. So if you enter your birthday you get your exact age in years. You can even get the age as number with up to 18 places after the decimal separator but I heard that this is a bit uncommon by the age of a person . . .

---

*This document corresponds to calcage v0.90, dated 2012/09/09.

2 Usage

Just load the package placing

> `\usepackage{calcage}`

in the preamble of your LaTeX 2_ε source file.

You can also give optional package options to change the default behavior. But you can not give values (see Bugs). So

> `\usepackage[presision]{calcage}`

works and sets presision to 3. But "presision=3" is going to fail . . .

\calcage The macro `\calcage` [⟨*key values*⟩] {⟨*year*⟩} {⟨*month*⟩} {⟨*day*⟩} takes a date and typesets the difference to the current date in years.

year, month, day

There are some optional parameters. For example you can adjust the current date for calcage with the keys "*year*", "*month*" and "*day*". You don't have to

precision specifier all of them. A subset is also possible.

Another useful parameter is "*precision*". With this switch you can specify how many places after the decimal separator you would like to see. The default precision is 0 which means that `\calcage` will typeset the age in years as integer.

positive If you omit a value for "precision" then the precision will be 3.

The macro `\calcage` can also take a date which is in the future. The default behavior in this case is a negative number but you can make the number positive

printyear with the "*positive*" boolean switch.

The next parameter I would like to show you is the "*printyear*" boolean switch. With this switch you can specify if `\calcage` should add "year" or rather "years"

yearsuffix after the year. This is the default setting.

For the previous parameter you may need the possibility (at least in the German language) to add a single character to the plural word. For example

> `Albert Einstein wurde vor \calcage[yearsuffix]{1879}{03}{14}`
> `geboren.`
> Albert Einstein wurde vor 133 Jahren geboren.

To do this you can use the boolean switch "*yearsuffix*". By default this switch is

numberstring false.

And last but not least there is the boolean switch "numberstring" which is true by default. This means that the typesetting process of an integer is done by the package fnumprint which will typeset the word of a number instead of the Arabic number if the value of the number is between 0 and 12. If you don't like this behavior you can set the boolean to false with "numberstring=false".

3 Bugs

I tried to implement the possibility to change the default behavior as package option in a nice way but with the current implementation there is a problem with the values. If you give a key with value than the `\setkeys` macro which gets these parameters as macro fails. I already tried `\expandafter` so if anyone knows where the problem is or has a solution feel free to contact me.

I think there is a little inaccuracy in the conversion from day to years because my implementation just removes the leap years before dividing but in my tests this seems to be more theoretically.

4 Examples

```
\calcage{2010}{09}{10} \\
\calcage{2009}{11}{09}
```

one year
two years

```
Donald Knuth is \calcage{1938}{01}{10} old. \\
Donald Knuth will be 100 years old in
   \calcage[positive, precision]{2038}{01}{10}. \\
Albert Einstein died at the age of
   \calcage[year=1955, month=04, day=18]{1879}{03}{14}. \\
Age of Linus Torvalds in years: \calcage[printyear=false]{1969}{12}{28} \\
Linus Torvalds is \calcageDays old.
```

Donald Knuth is 74 years old.
Donald Knuth will be 100 years old in 25.336 years.
Albert Einstein died at the age of 76 years.
Age of Linus Torvalds in years: 42

5 Implementation

This package depends on these packages.
```
1 \RequirePackage{fnumprint}[2012/08/27]
2 \RequirePackage{
3   datenumber,
4   fp,
5   calc,
6   xkeyval,
7   kvoptions,
8   xifthen,
9 }
```

5.1 Declaring the options

```
10 \DeclareStringOption{year}
11 \DeclareStringOption{month}
12 \DeclareStringOption{day}
13 \DeclareStringOption{precision}[3]
14 \DeclareBoolOption{positive}
15 \DeclareBoolOption{printyear}
16 \DeclareBoolOption{yearsuffix}
17 \DeclareBoolOption{numberstring}
```

To test if all parameters are valid the marco `\ProcessLocalKeyvalOptions*` is expanded to ensure this before leaving the preamble. This is the only purpose for the `\ProcessLocalKeyvalOptions*` in this case.

```
18 \ProcessLocalKeyvalOptions*
```

The next source code line creates a new macro which captures the parameters to use them as default options for the macro `\calcage`. If this macro is set to "precision=4" for example the `\setkeys` will fail.

```
19 \edef\calcage@options{\@ptionlist{\@currname.\@currext}}
20 %% ^^A \renewcommand{\calcage@options}{precision=4,printyear=false}
21 %% ^^A Package xkeyval Error: 'precision=4' undefined in families 'calcage'.
```

5.2 Language selection

Here comes the language selection part. The counter is all ready set by the fnumprint so I use it's value also in this package because I can rely on this counter.[1]
If you prefer another language than English or German you can redefine these macros.

```
22 \ifcase\value{fnumprint@language}\or
```

If fnumprint@language is equal 1

```
23   \newcommand{\calcage@yearWord}{Jahr}
24   \newcommand{\calcage@yearPluralSuffix}{e}
25   \newcommand{\calcage@yearSuffix}{n}
26 \or
```

If fnumprint@language is equal 2

```
27   \newcommand{\calcage@yearWord}{year}
28   \newcommand{\calcage@yearPluralSuffix}{s}
29   \newcommand{\calcage@yearSuffix}{}
30 \fi
```

5.3 Macro definition

Some necessary LaTeX counters are declared here before creating the `\calcage` macro.

```
31 \newcounter{calcage@today}\newcounter{calcage@ageindays}
32 \newcounter{calcage@myyear}\newcounter{calcage@leapyears}
```

---

[1]I am also the maintainer of the fnumprint package . . .

\calcage And now comes the important part – the definition of `\calcage`. First the `\setkeys` macro is expanded. It sets the default options then the package options get the opportunity to overwrite these settings and finally the macro options get the same possibility.

```
33 \newcommand{\calcage}[4][]{%
34   \setkeys{calcage}{precision=0, positive=true, printyear=true,
35     yearsuffix=false, numberstring=true,
36     year=\the\year, month=\the\month, day=\the\day, \calcage@options, #1}%
37   \setmydatenumber{calcage@today}{\calcage@year}{\calcage@month}{\calcage@day}%
38   \setmydatenumber{calcage@ageindays}{#2}{#3}{#4}%
```

Now comes the tricky part which are the leap years . . .
My first implementation worked with dividing by 365.2425 but this was not perfect.
So the current implementation counts how many leap years are between the birth year and the current year and subtracts (or adds[2]) this number.

```
39   \setcounter{calcage@myyear}{#2}%
40   \setcounter{calcage@leapyears}{0}%
41   \ifthenelse{\equal{#2}{\calcage@year}}{}{%
42     \ifthenelse{\value{calcage@myyear}<\calcage@year}{%
```

If the birth year is in the past.

```
43       \loop%
44         \stepcounter{calcage@myyear}%
45         \ifleapyear{\thecalcage@myyear}\stepcounter{calcage@leapyears}\fi%
46         \ifnum\value{calcage@myyear}<\calcage@year%
47       \repeat%
48     }{%
```

If the birth year is in the future.

```
49       \loop%
50         \ifleapyear{\thecalcage@myyear}\addtocounter{calcage@leapyears}{-1}\fi%
51         \addtocounter{calcage@myyear}{-1}%
52         \ifnum\value{calcage@myyear}>\calcage@year%
53       \repeat%
54     }%
55   }%
56   \setcounter{calcage@ageindays}{\value{calcage@today}
57     - \value{calcage@ageindays} - \value{calcage@leapyears}}%
58   \ifthenelse{\boolean{calcage@positive} \AND \value{calcage@ageindays} < 0}{%
59     \setcounter{calcage@ageindays}{\value{calcage@ageindays} * -1}%
60   }{}%
```

Because we know that every year has now exactly 365 day we can divide by that.

```
61   \FPdiv\calcage@age{\thecalcage@ageindays}{365}%
```

The next step is to truncated the age. There is no rounding. I implemented it this way because if the age would be rounded then there would be a wrong result in cases like $x.999$ which would become $x + 1$ if "precision=0" and that is probably not what you want . . .

```
62   \FPtrunc\calcage@age{\calcage@age}{\calcage@precision}%
```

---

[2]If the birth year is in the future.

The last thing to do is to typeset the age which is implemented by the next few lines of code.

```
63    \ifthenelse{\equal{\calcage@precision}{0} \AND \boolean{calcage@numberstring}}%
64      {\fnumprint[ein]{\calcage@age}}{\numprint{\calcage@age}}%
65    \ifthenelse{\boolean{calcage@printyear}}{%
66      ~\calcage@yearWord%
67      \ifthenelse{\equal{\calcage@age}{1}}{}{%
68        \calcage@yearPluralSuffix%
69        \ifthenelse{\boolean{calcage@yearsuffix}}{\calcage@yearSuffix}{}%
70      }%
71    }{}%
72 }
```

That's it.

```
73 \endinput
```

Change History

0.90

    General: Initial version   1

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.