The totcount package*

Vasileios Koutavas
Vasileios.Koutavas@cs.tcd.ie

14 April 2009

Abstract

This is the documentation for the totcount package, a package for computing and displaying the value that was assigned last to counters inside a document. With this package one may count the total number of elements (e.g. sections, pages, citations, etc.) in a LaTeX document by simply using standard LaTeX counters.

Contents

1 Introduction

Referring to the total number of sections, pages, citations, list items, or anything else in a document can be difficult to achieve. The difficulty comes when the reference is *before* the definition of all the elements that need counting. In this case, abandoning the manual way requires the use of auxiliary files in much the same way that LaTeX uses for references and citations.

Special packages like totpages [4] and lastpage [2] handle this issue for the total number of pages, but, to the best of my knowledge, there is no package that makes it easy to print the total number of arbitrary elements.

The TotCount package hopefully fills this gap. It enables the computation and display of the number that was assigned last to a counter inside a document (usually the maximum value of the counter). The user just has to include the TotCount package, and *register* a counter as a "total counter". Then, getting the

*Available at CTAN:macros/latex/contrib/totcount/.

maximum value of that counter is as easy as calling the macro `\total{`⟨*counter*⟩`}` at the desired place, and running LaTeX twice.

As an early example, the commands

```
⟨*doc⟩
\regtotcounter{section}\total{section}
⟨/doc⟩
```

inform the reader that there are 7 sections in this document. With only slightly more work (since there is no convenient counter already defined) we can write that this document has 4 citations, which it mentions 6 times:

```
⟨*doc⟩
\newtotcounter{citnum}
\def\oldbibitem{} \let\oldbibitem=\bibitem
\def\bibitem{\stepcounter{citnum}\oldbibitem}
\total{citnum}

\newtotcounter{citesnum}
\def\oldcite{} \let\oldcite=\cite
\def\cite{\stepcounter{citesnum}\oldcite}
\total{citesnum}
⟨/doc⟩
```

2 User Interface

`\regtotcounter` To display the maximum value of a counter we first need to *register* that counter as a "total counter". We can do this by calling the macro `\regtotcounter{`⟨*counter*⟩`}` on an existing counter. We can register a counter in the preamble or inside the document, but in any case before calling `\total` or `\totvalue` on that counter.

This macro will use the main auxiliary file to get the total number of the counter at the second time we run LaTeX on the document.

To use a file other than the main auxiliary file we need to pass an option to the macro: `\regtotcounter[auxfile=`⟨*file*⟩`]{`⟨*counter*⟩`}`. This way the last value of ⟨*counter*⟩ will be stored in ⟨*file*⟩. This macro will also input ⟨*file*⟩ the second time that LaTeX runs (and every time after that) to get the right total value of the counter. The same auxiliary file can be used for many total counters and multiple auxiliary files can be used within the same document.

`\newtotcounter` The macro `\newtotcounter` is a shorthand for creating a new counter and registering it as a total counter. Just like `\regtotcounter`, we can pass an `auxfile` option to this macro to make it use an alternative auxiliary file: `\newtotcounter[auxfile=`⟨*file*⟩`]{`⟨*counter*⟩`}`.

`\total` To print the maximum value of ⟨*counter*⟩ we can call the macro `\total{`⟨*counter*⟩`}`. The first time LaTeX runs on the document this macro will display the symbols "??" and write the message "`Rerun to get correct total counts`" in the terminal. The second time LaTeX runs on the same document (and every time after that) the macro will display the correct total count of ⟨*counter*⟩.

`\totvalue` We can obtain the numeric total value of ⟨*counter*⟩ (in contrast to printing it using `\total`) by calling the macro `\totvalue{`⟨*counter*⟩`}`. This is useful, for

example, when we want to test the total value of the counter (see example in Section 6). The first time that LaTeX runs on a document, where the total counts are not computed yet, this command returns `-1`.

`\usetotcountfile` Sometimes we might want to use the total counters of an auxiliary file without recomputing (and overriding) their value. This is especially useful when the total counters have been computed by running LaTeX on a different document. In this case, instead of registering a total counter with one of the commands `\regtotcounter` and `\newtotcounter` we can simply use the auxiliary file that contains the values of the desired total counters by calling the macro `\usetotcountfile{`⟨*file*⟩`}`. This functionality is the main reason why total counters can be stored in alternative auxiliary files.

`\newcounter` `\addtocounter` `\stepcounter` `\setcounter` `\value` Since this package uses regular LaTeX counters, we can use all the usual commands to define, set, increment, and get the current (*not the total*) value of counters. See The LaTeX Companion [3] for an explanation of how to use LaTeX counters, and Section 6 for sample uses of these macros.

3 Installation

To get the package file run

```
latex totcount.ins
```

Then copy the `totcount.sty` file in a directory accessible by (pdf)LaTeX and re-hash the latex tree.

To compile the documentation run

```
pdflatex totcount.drv
makeindex -s gind.ist totcount
makeindex -s gglo.ist -o totcount.gls totcount.glo
pdflatex totcount.drv
```

To run the example do

```
pdflatex totcount-ex.tex
pdflatex totcount-ex.tex
```

4 Required Packages

The TotCount package requires the keyval package [1].

5 Implementation

This section contains the source code of `totcount.sty`.

```
⟨*sty⟩
\ProvidesPackage{totcount}
  [\filedate \space v\fileversion \space package for getting%
   the total value of LaTeX counters]
```

Import of the keyval package [1]:

```
\RequirePackage{keyval}
```

`\newtotcounter` Creates a new counter and registers it as a total counter. This is the top-level dispatch of the macro, depending on whether there is an optional argument or not.

```
\def\newtotcounter{%
  \@ifnextchar[\newtotcounter@newaux\newtotcounter@mainaux}
```

`\newtotcounter@newaux` This is the version of the `\newtotcounter` macro that uses a separate auxiliary file. It first creates the counter (second argument) and then calls the macro `\regtotcounter`.

```
\def\newtotcounter@newaux[#1]#2{%
  \newcounter{#2}%
  \regtotcounter[#1]{#2}%
}
```

`\newtotcounter@mainaux` This is the version of the `\newtotcounter` macro that uses the main auxiliary file. It first creates the counter (argument) and then calls the macro `\regtotcounter`.

```
\def\newtotcounter@mainaux#1{%
  \newcounter{#1}%
```

Register the counter:

```
  \regtotcounter{#1}%
}
```

`\regtotcounter` Registers a counter as a total counter. This is the top-level dispatch of the macro, depending on whether there is an optional argument or not.

```
\def\regtotcounter{%
  \@ifnextchar[\regtotcounter@newaux\regtotcounter@mainaux}
```

The following is the definition of the `auxfile` key for specifying an alternative auxiliary file when calling the macro `\regtotcounter` (see [1]).

```
\define@key{totcounter}{auxfile}{\def\this@auxfile{#1}}
```

`\regtotcounter@newaux` This is the version of the `\regtotcounter` macro that uses a separate auxiliary file. The auxiliary file is passed as a first argument in the form of a key–value pair `[auxfile=`⟨*file*⟩`]`, and the counter to be registered is passed as the second argument.

```
\def\regtotcounter@newaux[#1]#2{%
  \setkeys{totcounter}{#1}%
```

Try to load the contents of the file:

```
  \InputIfFileExists{\this@auxfile}{}{}%
```

Make sure that the auxiliary file is open; LaTeX will close it at the end:

```
  \expandafter\ifx\csname \this@auxfile @open\endcsname\relax%
    \expandafter\gdef\csname \this@auxfile @open\endcsname{}%
    \expandafter\newwrite\csname \this@auxfile @stream\endcsname%
```

```
    \immediate\expandafter\openout%
      \csname \this@auxfile @stream\endcsname=\this@auxfile%
  \fi%
```

Create a new counter holding the total number of the actual counter:

```
  \expandafter\ifx\csname c@#2@totc\endcsname\relax%
     \newcounter{#2@totc}%
     \setcounter{#2@totc}{-1}%
  \fi%
```

At the end of the document write code in the auxiliary file to update the total counter with the value of the actual counter:

```
  \AtEndDocument{%
     \def\sp{ }%
     \immediate\expandafter\write%
     \csname \this@auxfile @stream\endcsname{%
       \string\expandafter\string\ifx%
         \string\csname\sp c@#2@totc\string\endcsname\string\relax%
       \string\newcounter{#2@totc}%
       \string\fi%
       \string\setcounter{#2@totc}{\number\value{#2}}%
     }%
  }%
}
```

`\regtotcounter@mainaux` This is the version of the `\regtotcounter` macro that uses the main auxiliary file. The counter to be registered is passed as the second argument.

```
\def\regtotcounter@mainaux#1{%
```

Create a new counter holding the total number of the actual counter:

```
  \expandafter\ifx\csname c@#1@totc\endcsname\relax%
     \newcounter{#1@totc}%
     \setcounter{#1@totc}{-1}%
  \fi%
```

At the end of the document write code in the auxiliary file to update the total counter with the value of the actual counter:

```
  \AtEndDocument{%
    \def\sp{ }%
    \immediate\write\@mainaux{%
      \string\expandafter\string\ifx%
        \string\csname\sp c@#1@totc\string\endcsname\string\relax%
      \string\newcounter{\string #1@totc}%
      \string\fi%
      \string\setcounter{\string #1@totc}{\number\value{#1}}%
    }%
  }%
}
```

`\total` Prints the total value of a registered total counter that is passed as argument. If the total value is yet to be computed (at the first time LaTeX runs on the

document) then its value is `-1` and the output of the command is ??.

```
\newcommand\total[1]{%
  \def\tmp@val{\value{#1@totc}}%
  \ifnum\tmp@val=-1%
    \message{Rerun to get correct total counts}%
    $??$%
  \else%
    \number\value{#1@totc}%
  \fi%
}
```

`\totvalue` Returns the numeric total value of a registered total counter that is passed as argument. Note that if the counter's value is not yet computed (at the first time LaTeX runs on the document) this macro returns `-1`.

```
\newcommand\totvalue[1]{\value{#1@totc}}
```

`\usetotcountfile` Inputs an auxiliary file that should contain total-counter definitions. It outputs a warning message in the terminal if the file doesn't exist. This command should be used *instead* of registering a counter.

```
\newcommand\usetotcountfile[1]{%
  \InputIfFileExists{#1}{%
    \message{TotCount inputs file '#1'}%
  }{%
    \message{TotCount Warning: File '#1' does not exist!}%
    \message{\space\space\space\space\space\space\space\space\space%
             \space\space\space\space\space\space\space\space\space%
             Ignoring \string\usetotcountfile{#1}.}%
  }%
}
⟨/sty⟩
```

6 Simple Example

The following is a simple example of how to use the functionality of the TotCount package. This example can be found in the file `totcount-ex.tex`, after running LaTeX on the `totcount.ins` file (see Section 3).

```
⟨*ex⟩
\ProvidesFile{totcount-ex.tex}
      [\filedate \space v\fileversion \space%
       test file for the TotCount package]
\documentclass{article}
```

Import the package

```
\RequirePackage{totcount}
```

Comment out the following line to use the previously stored total counts in 'mycount.aux' (make sure it exists!) instead of computing them again:

```
\def\computemycount{}
```

```
\expandafter\ifx\csname computemycount\endcsname\relax
  \usetotcountfile{mycount.aux}
  \newcounter{mycount} % so that increments of this counter don't break
\else
  \newtotcounter[auxfile=mycount.aux]{mycount}
\fi
```

Register the section and page counters as a total counters:

```
\regtotcounter{section}
\regtotcounter{page}

\begin{document}
```

A total counter can be declared/registered anywhere:

```
\newtotcounter{yourcount}

\section{Total Counts}
```

Increment counters as usual:

```
\addtocounter{mycount}{5}
\addtocounter{yourcount}{5}
```

Print the value of a total counter by using `\total`:

```
The total value of the counter {\tt mycount} is \total{mycount} (should
be 10), and the total value of the counter {\tt yourcount} is
\total{yourcount} (should be 17).

```

Get the numeric value of a total counter by using `\totvalue`:

```
This document has \total{section}
\ifnum\totvalue{section}=1 section \else sections \fi
and \total{page}
\ifnum\totvalue{page}=1 page. \else pages. \fi

```

The effect of incrementing the counters will be visible the second time LaTeX runs:

```
\addtocounter{yourcount}{5}
\addtocounter{mycount}{5}
\addtocounter{yourcount}{7}

\section{A Section}
\section{Another Section}
\section{Yet Another Section}

\end{document}
```

⟨/ex⟩

7 Acknowledgments

Many thanks to Christoforos Moutafis for pointing me to the right direction when I first started looking for a way to display the maximum value of counters.

References

[1] David Carlisle, *The keyval package*, LaTeX 2ε package.
CTAN:macros/latex/required/graphics/keyval.dtx

[2] Jeff Goldberg, *The lastpage package*, LaTeX 2ε package.
CTAN:macros/latex/contrib/lastpage/

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin, *The LaTeX Companion*, Addison-Wesley, 1994.

[4] Wilhelm Müller, *The totpages package*, LaTeX 2ε package.
CTAN:macros/latex/contrib/totpages/

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.

Change History

v1.0

General: First release 1