# mailmerge: repeat amount of text merging fields

Miguel Vinícius Santini Frasson

2009–09–23 version 1.0

## Contents



## 1 Introduction

The package mailmerge provides an interface to mail merge or other such documents, where a body (or amount of text) is to be repeated, with some tags replaced by respective values in each repetition.

## 2 Usage

In short, one needs to:

1. declare field names,
2. define the repetition and
3. provide entries, each one being a set of values for the fields.

It is important to define the repetition with `\mailrepeat` *before* provide any entry. We present the usage in more detail now.

\mailfields

1. Declare the names of the fields, with the command
   \mailfields{*name1*,*name2*,...}
   The order declared here is the order in which the values must be provided in the parameter of \mailentry (see below.)

\mailrepeat
\field

2. The block of text to be repeated is given as the parameter of macro \mailrepeat as in
   \mailrepeat{*... text to be repeated...*}
   Inside the text to be repeated, which can be several paragraphs long, each instance of \field{*name*} is replaced by the respective value, this for each entry.

\numberoffields
\numberofentries
\entrynumber

The command \numberoffields is replaced by the number of fields and \numberofentries is replaced by the number of entries, but it is necessary two runs to update the value. The command \entrynumber is replaced by the current entry number. This can be used, for example, in determining if current entry has an odd or even number.

For conditional text, depending on field value, one can compare value's contents, for example, with the \ifthenelse command provided by the ifthen package.

\mailentry

3. For each entry, provide a command
   \mailentry{*value1*,*value2*,...}
   where *value1* is the value of field named *name1* , *value2* is the value of field named *name2*, and so on. The values can be several paragraphs long. If the value has a comma with it, better enclose the value between curly brackets, like in

```
\mailfields{name,friends,drives}
\mailentry{John,{Bart,Robert},yes}
\mailentry{Michael,{Jean,Phillip,Maria},no}
```

\mailnewdata

4. In case of intention of use of several databases in the same file, use the command \mailnewdata, which resets entries. New uses of \mailfields overhide previous field names.

# 3 Example

```
\usepackage{ifthen,mailmerge}

% \ifequal{A}{B}{what if A=B}{what if A<>B}
\newcommand{\ifequal}[2]{\ifthenelse{\equal{#1}{#2}}}

\mailfields{name,friends,drives}

\mailrepeat{\section*{\field{name}'s profile}
```

```
    \field{name} has
    \ifequal{\field{friends}}{}
      {no friends}
      {\field{friends} as friends}.
    \ifequal{\field{drives}}{yes}{Drives.}{Doesn't drive.}

  (entry \entrynumber\ of \numberofentries)

  % \newpage optional
  }

  \mailentry{John,{Bart and Robert},yes}
  \mailentry{Sara,{Jean, Phillip and Maria},no}
  \mailentry{Edward,,yes}
```

what produces the following output:

## John's profile

John has Bart and Robert as friends. Drives.
(entry 1 of 3)

## Sara's profile

Sara has Jean, Phillip and Maria as friends. Doesn't drive.
(entry 2 of 3)

## Edward's profile

Edward has no friends. Drives.
(entry 3 of 3)

# 4 Idea of implementation

- Each new data initialization (call od `\mailnewdata`) defines a tag command with values 'a', 'aa', 'aaa', and so on, saved in command `\MAILMcurrtag`.
- A `\mailfields` saves field names to commands `\MAILMfieldI`, `\MAILMfieldII`, `\MAILMfieldIII`, etc (ending with upcase roman numbers), and at the end, saves in the `.aux` file the number of fields in a command determined by the data tag.
- The `\mailrepeat` command just saves its parameter to a command.

- Each \field{*name*} is expanded to a command \MAILMthefield*name*.
- \numberoffields expands to a command whose name is composed by MAILMnumberfields plus "tag". The same for \numberofentries.
- Each \mailentry defines the comands \MAILMthefield*nameN* to expand to *valueN*, using \MAILMfield*N-in-roman* to compose the name of the command and expand an entry of the repetition command. Now, when the \field, \numberoffields and \numberofentries are expanded, they have ppropriated values.

# 5 Code

Identidication of the package and use of the package ifthen.

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{mailmerge}[2009/09/23 v1.0 repeat amount of text merging fields]
\RequirePackage{ifthen}
```

## 5.1 (Re)initialization

Reset counters and update tag command \MAILMcurrtag. Inicialize once.

```
\def\MAILMcurrtag{}
\newtoks\MAILMtok
\newcounter{MAILMcount}% aux counter
\newcounter{MAILMentry}%
\newcommand{\mailnewdata}{%
  \setcounter{MAILMentry}{0}%
  \xdef\MAILMcurrtag{a\MAILMcurrtag}}
\mailnewdata
```

## 5.2 Saving the repetition in a command

\mailrepeat

```
\newcommand{\mailrepeat}[1]{\gdef\MAILMrepetition{#1}}
```

## 5.3 Restoring data from aux file

\MAILMsetnumfields
\MAILMsetnumentries
\numberoffields
\numberofentries

Commands that extract info from aux file (saved by \mailfields and \mailentry) and commands that use this information when repeting.

```
\newcommand{\MAILMsetnumfields}[2]
  {\expandafter\xdef\csname MAILMnumberoffields#1\endcsname{#2}}
\newcommand{\MAILMsetnumentries}[2]
  {\expandafter\xdef\csname MAILMnumberofentries#1\endcsname{#2}}
\newcommand{\numberoffields}
  {\csname MAILMnumberoffields\MAILMcurrtag\endcsname}
\newcommand{\numberofentries}
  {\csname MAILMnumberofentries\MAILMcurrtag\endcsname}
```

## 5.4 Saving field names

```
\newcommand{\mailfields}[1]{%
  \setcounter{MAILMcount}{0}%
```

Initiate a loop, where for each entry, the command `\MAILMaux` is set to entry name.

```
  \@for\MAILMaux:=#1\do{%
```

The next line is to extract white space after comma.

```
    \edef\MAILMaux{\expandafter\@firstofone\MAILMaux\@empty}%
```

Define command `\MAILMfield`*Roman-number*.

```
    \stepcounter{MAILMcount}%
    \expandafter\edef\csname MAILMfield\Roman{MAILMcount}\endcsname
         {\MAILMaux}%
    \edef\numberoffields{\arabic{MAILMcount}}%
  }%
```

Write to aux `\MAILMsetnumfields{tag}{num}`

```
  \immediate\write\@mainaux{\string\MAILMsetnumfields
    {\MAILMcurrtag}{\theMAILMcount}}%
}
```

## 5.5 Repeating within an entry

`\field` Defining `\field`, that expands to `\MAILMthefield`*name*.

```
\newcommand{\field}[1]{\csname MAILMthefield#1\endcsname}
```

`\mailentry` Each entry increments counter (reset in initialization), saves entry number to aux, saves each value to the command expanded by `\field` and expand the repetition, stored in `\MAILMrepetition`. The implementation is similar to `\mailfields`.

```
\newcommand{\mailentry}[1]{%
  \stepcounter{MAILMentry}%
  \edef\entrynumber{\theMAILMentry}%
  \immediate\write\@mainaux{\string\MAILMsetnumentries
    {\MAILMcurrtag}{\theMAILMentry}}%
  \setcounter{MAILMcount}{0}%
  \@for\MAILMentryfield:=#1\do{%
     \MAILMtok=\expandafter{\MAILMentryfield}%
     \stepcounter{MAILMcount}%
     \expandafter\long\expandafter\edef
        \csname MAILMthefield%
           \csname MAILMfield\Roman{MAILMcount}\endcsname
        \endcsname {\the\MAILMtok}%
  }%
  \MAILMrepetition
}
```

# 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.