FRISTI

The Foundation for Responsible Info Stuffing Inventions

Herman Haverkort

`hh@fgbbs.iaf.nl`

September 1996

Abstract

This article gives an impression of my way of designing and typesetting very small telephone and address lists, birthday calendars etc. Design considerations and typesetting tricks are presented. The latter include macros which define the sheets (from A4 to credit card and even key fob size), macros to process data records (typically specified in a separate file) in various ways, and macros to stuff and stow data in very limited space.

Keywords: small, list, directory, calendar, key fob, credit card, stuffing

Preface

The macros which are presented in this article are not treated in full detail, some are only vaguely mentioned. This article is not meant to be a manual, my goal is to give you an impression of what can be accomplished with LaTeX in the field of Responsible Info Stuffing Inventions. All macros presented in this article are part of the hhfristi package. If you are interested in getting the macros and more thorough explanation, I will happily provide them on request.

The development of all that is presented here, originated in the shortcomings of an ordinary membership list of an extraordinary youth orchestra. The orchestra consists of about fifty members, which are listed yearly on an A4 sized address list, which is handed out to all members. I found that the address lists were ill suited to quite a big part of their daily use, and I wanted to make better lists. Moreover, I wanted to typeset them automatically. The design principles presented in this article are mostly based on the practical problems I encountered; they are not the result of a thorough study of lists in general and therefore may not always be as valuable as they are to me.

Choosing the size of the list

An A4 size list is easy to produce, since it is the standard paper size for typewriters, copying machines etc.[1], and it is very well suited to be stored in an archive, which typically is an ordner, or a box, containing a lot more A4's which the list can be stored nicely in between. However, an A4 size list may not be such a good list to have at hand in varying circumstances. To stuff an A4 in a wallet, you must fold it at least three times. On the folds the text will wear off, and eventually the list will tear. Besides, an A4 size sheet of paper folded three times will be almost one millimeter thick; having a few of those lists in your wallet will contribute quite a lot to uncomfortable thickness. Furthermore, big lists, especially wide lists, are often hard to read correctly, as will be explained further on.

For always-at-hand lists I prefer a page size of A7 maximum. One can construct bigger lists, but then I recommend doing it in such a way that the list can be folded to A7 size or less, without having the folds cut the text. In table 2 you will find a summary of the list sizes supported by the hhfristi package.

The hhfristi list sizes and shapes can be selected using the `fristiform` environment. Text inside this environment is formatted to fit the specified shape and size, and consequently output in one or two boxes (for one and two sided lists). These boxes can be printed next to each other by including the `fristiform` environment in a `\hbox`, or on top of each other by using a `\vbox`. The `fristiform` environment requires one argument which is a list of options separated by commas. Simple options consist of a single keyword; value options consist of a keyword followed by the '=' character and some kind of value. The options supported are listed in tables 1 and 2.

Selecting font sizes

To make it easier to experiment with different font sizes I developed a macro `\hhfrsizes`. A typical use of this macro reads `\hhfrsizes{8pt}{1.05}`. It sets `\normalsize` to 8pt, sets all other font size macros (`\scriptsize`, `\small`, `\large` etc.) to appropriate values relative to the `\normalsize`, and sets the baseline stretch to 1.05. All font sizes calculated are rounded to standard values between 5pt and 24.88pt (10pt $\times 1.2^5$).

[1] At least it is in the Netherlands; substitute a similar size for other countries in this sentence.

Simple options		
keyword, selected by default	**alternative option**	**description**
`landscape`	`portrait`	to be used after size or width and height options; landscape has no effect; portrait switches width and height values
`row`	`stack`	determines if pages are to be arranged as a row (vertical folds and turn-over axis) or as a stack (horizontal folds and turn-over axis)
`noeye`		specifies that no space for a perforation is to be created
`nooutline`	`outline`	determines wether or not the outline of the list shape is to be drawn

Table 1: Simple options for the fristiform environment

Value options			
keyword	**values**	**default**	**decription**
`size`	see descr.	A6	legal values are for example: **C** credit card size (85.5mm × 53.8mm); **A6** A0 size, halved six times by shortening its longest side (148.7mm × 105.2mm); **A5/2** A5 size, halved by shortening its *shortest* side (210.5mm × 74.3mm); **A4/2/3** A4 size, with its shortest side divided by two and its longest side divided by three (105.2mm × 99.1mm); Please note that any positive integers can be used instead of the values used above; any basic size identifier can be used instead of the 'A' and 'C' used above. Size identifiers 'A', 'B', 'C', 'E' (executive paper) and 'L' (letter paper) are predefined with macro calls like `\hhfr@basicsize {A}{1189.21mm}{841.90mm}`; other identifiers can be defined similarly; the size options always sets width and height so that the width is the longest side
`width`	dimensions	148.7mm	sets the total width of the list
`height`	dimensions	105.2mm	sets the total height of the list
`columns`	1, 2, ...	1	selects number of columns per page
`colsep`	dimensions	4pt	sets distance between columns
`colseprule`	dimensions	0.4pt	sets width of rule between columns
`folds`	0, 1, ...	0	selects how many folds the list should have
`fold`	dimensions	15pt	sets the amount of unused space around folds
`hem`	dimensions	6pt	if the number of folds is two or more, then strips parallel to the folds are created at the ends of the list; the width of the strips is two times the hem value (the strips can be doubled up to provide a kind of 'handles' which can be gripped to unfold the list)
`sides`	1, 2	2	selects one or two sided list
`outerhmar`	dimensions	6pt	sets the horizontal distance between border and text (outer left and right margin)
`innerhmar`	dimensions	6pt	sets the horizontal distance between fold space and text (left and right margin around folds; the total horizontal distance between two pages equals the fold plus two times the innerhmar)
`outervmar`	dimensions	3pt	anologous to outerhmar
`innervmar`	dimensions	3pt	anologous to innerhmar
`eyepos`	l, r, t, b	not set	determines where space for a perforation should be created
`eye`	dimensions	20pt	sets the diameter of the space taken by the perforation (this space is independent of the perforation outline which is drawn)
`eyemar`	dimensions	0pt	sets the extra margin between the perforation space and the text (this margin is added to the outerhmar or outervmar used)
`corner`	dimensions	10pt	sets the radius of the rounded corners
`seal`	dimensions	0pt	the seal value represents the extra margin for sealing the laminate; it is subtracted twice from the total width and height
`fontsize`	dimensions	not set	sets fristi font sizes to the specified value and sets baselinestretch to one using `\hhfrsizes`
`stretch`	positive v.	not set	sets the baselinestretch to the specified value

Table 2: Value options for the fristiform environment

Designing the lay-out of a list

While lay-outing an address list, it is important to realize how the list will be used. A typical list contains one line for every person who is on it, while each line contains a number of fields of information about a person. Information can be read vertically (e.g. reading the residence field of every person) or horizontally (e.g. reading all the information about one person). In general only the names (family or first names, depending on the application) are read vertically, so names should be listed on a vertical line, below each other. Thus it is easy to read vertically and to use the alphabetical order to find a particular name. When that particular name is found, information about that person is read horizontally. To ease this horizontal reading it is important that the information about one person is on a horizontal line which the eye can easily follow. In table 3 this is clearly not the case. The horizontal lines are unclear, so that it is hardly possible to find correctly someone's telephone number without using a ruler or moving your finger horizontally along the paper, from name to number.

In table 4 the horizontal lines are much clearer because all information about one person is listed directly next to each other, without large white gaps in between. Of course vertical reading of other fields than the name is very difficult on this list (which is irrelevant in many practical applications), and the table does not look very neat.

Table 5 shows a compromise which looks better to some folks, and enables vertical reading of all fields. The horizontal lines are elucidated by connecting rules between the fields, and by additional separating rules after each fifth line.

In general shorter horizontal lines give better readability (this is also a reason why big lists are not always better than small ones). In table 6 this is reckoned with in various ways, although this list is also a compromise between various readability and aesthetic demands. In this case the information which is needed most is the telephone number; therefore it is typeset close to the name. Next is the postcode, which I often do not know by heart, and finally the address, which is put at the end because I often do know it by heart. Extraordinary long names, like Victoria van Asch van Wijck, are typeset (partly) in a smaller font. This injures the readability of that particular name, but if I had enlarged the gap between name and phone number on all other lines instead, I would have injured the readability of the whole list. Above that I would be forced to create space by choosing a smaller (less readable) font for the whole list.

Stuffing techniques

When making the list in table 6, I intended to stuff a lot of information on a small sheet of paper in a very well readable fashion, so: in a font size as large as possible, using abbreviations only when needed. I used the following tricks to stuff info:

- printing extraordinary long names in a smaller font;
- printing information which is less frequently needed in a smaller font;
- raising abbreviated prefixes etc. (thus the period and the space which normally follow an abbreviation can be omitted);
- using different fonts for different fields (thus making it possible to put different fields very close to each other, without making the optical distinction between them unclear; see, for example, the postcodes and phone numbers);
- omitting the zero (the 'interlocal access code') that precedes the area codes;
- omitting area codes which equal that of most members, leaving more space for printing the names;
- omitting the first digit of the postcode if it equals that of most members (in this case: all members);
- omitting residences (which I often know by heart, and if not, they can be deduced from the postcodes using the table which is printed on the head of the list), unless there is enough space left in the address field;

Of course, the last three tricks can only be applied for local societies etc.

Most of the above mentioned tricks I only want to use when compact typesetting is really necessary. Therefore I designed 'stuffers', macros which can act differently depending on the stuffing pressure. An example of such a macro is `\sml`. The macro `\sml` is defined using `\defstuffer {sml}{3}{#1#2#3}[2]{{\small #2}}`: `\sml` gets `{3}` arguments, `\sml` normally typesets them all (`{#1#2#3}`), but if the level of stuffing pressure is `[2]` or less (the lower the number, the higher the pressure), only the middle argument is typeset, and it is typeset small. Stuffing directions like `\lng`, `\sht` and `\abb` are used to choose between full typesetting and abbreviations, and in the case of residence names: between typesetting them and entering them in the postcode table.

In general stuffers are passive: they always act as if there were a lot of space. However, when using the macro `\hhfrsqueeze` they are useful. `\hhfrsqueeze {`*dimension*`}{`*text*`}` tries to stuff *text* in a `\hbox` which is at most *dimension* wide. It does so by experimenting with different settings of the level of stuffing pressure, so that stuffers contained in the *text* can have its use.

If the option `dutch` is used, macros for "van de", "straat" etc. are predefined, making use of appropriate stuffers.

I did not design a general framework for handling text which really does not fit yet. However I did handle a special case: the case of two items sharing partly the same information, for example two persons having the same address. A macro is provided which facilitates trying to typeset them both on one line, and if that turns out to be impossible, typesetting them each on a separate line after all. I will not treat this `\hhfrduo` macro in detail here because until now the only practical examples of its use are quite complicated.

Arts, Lieke	Gemertstraat 16	6844 HC	Arnhem	(026) 39 11 1 11
van Asch van Wijck, Victoria	Tunnelweg 4	6601 CW	Wijchen	(024) 66 16 9 57
Bezemer, Sem	Alkmaarsingel 230	6843 WR	Arnhem	(026) 6 84 628 6
Buddeke, Sarah	Sint Caeciliapad 35	6815 GM	Arnhem	(026) 48 40 17 2
van den Bijlaard, Hans	Laan van Klarenbeek 105	6824 JN	Arnhem	(026) 41 26 20 5
van den Bijlaard, Quirijn	West Breukelderweg 15	6721 MP	Bennekom	(0318) 1 21 75 6
Constandse, Arthur	Wielewaalstraat 5	6823 DA	Arnhem	(026) 9 48 92 99
Ehlert, Arvid	Gemertstraat 17	6844 HD	Arnhem	(026) 7 85 05 67
van Elden, Ariette	Prins Bernhardweg 18	6862 ZH	Oosterbeek	(026) 36 44 34 2
van Elden, Remelie	Prins Bernhardweg 18	6862 ZH	Oosterbeek	(026) 36 44 34 2
van Es, Xander	Haarlemweg 20	6843 AM	Arnhem	(026) 40 13 85 4
Fabels, Marc	Rijnkade 39c	6811 HA	Arnhem	(026) 4 90 4 008
van Ge en, Pim	Zwarteweg 3	6923 CK	Groessen	(0316) 7 27 9 39
Geurts, Karin	Mierlostraat 92	6844 DZ	Arnhem	(026) 41 16 30 0
van Gurp, Thomas	Annastraat 7	6862 CG	Oosterbeek	(026) 38 37 35 6
Haverkort, Herman	Zijpendaalseweg 17	6814 CB	Arnhem	(026) 35 16 7 23
Haverkort, Koen	Monnikensteeg 264	6823 AL	Arnhem	(026) 51 39 36 1

Table 3: An example of a list with unclear horizontal lines

Arts, Lieke Gemertstraat 16 6844 HC Arnhem (026) 39 11 1 11
van Asch van Wijck, Victoria Tunnelweg 4 6601 CW Wijchen (024) 66 16 9 57
Bezemer, Sem Alkmaarsingel 230 6843 WR Arnhem (026) 6 84 628 6
Buddeke, Sarah Sint Caeciliapad 35 6815 GM Arnhem (026) 48 40 17 2
van den Bijlaard, Hans Laan van Klarenbeek 105 6824 JN Arnhem (026) 41 26 20 5
van den Bijlaard, Quirijn West Breukelderweg 15 6721 MP Bennekom (0318) 1 21 75 6
Constandse, Arthur Wielewaalstraat 5 6823 DA Arnhem (026) 9 48 92 99
Ehlert, Arvid Gemertstraat 17 6844 HD Arnhem (026) 7 85 05 67
van Elden, Ariette Prins Bernhardweg 18 6862 ZH Oosterbeek (026) 36 44 34 2
van Elden, Remelie Prins Bernhardweg 18 6862 ZH Oosterbeek (026) 36 44 34 2
van Es, Xander Haarlemweg 20 6843 AM Arnhem (026) 40 13 85 4

Table 4: An example of a list with unclear vertical lines

Arts, Lieke	Gemertstraat 16	6844 HC	Arnhem	(026) 39 11 1 11
van Asch van Wijck, Victoria	Tunnelweg 4	6601 CW	Wijchen	(024) 66 16 9 57
Bezemer, Sem	Alkmaarsingel 230	6843 WR	Arnhem	(026) 6 84 628 6
Buddeke, Sarah	Sint Caeciliapad 35	6815 GM	Arnhem	(026) 48 40 17 2
van den Bijlaard, Hans	Laan van Klarenbeek 105	6824 JN	Arnhem	(026) 41 26 20 5
van den Bijlaard, Quirijn	West Breukelderweg 15	6721 MP	Bennekom	(0318) 1 21 75 6
Constandse, Arthur	Wielewaalstraat 5	6823 DA	Arnhem	(026) 9 48 92 99
Ehlert, Arvid	Gemertstraat 17	6844 HD	Arnhem	(026) 7 85 05 67
van Elden, Ariette	Prins Bernhardweg 18	6862 ZH	Oosterbeek	(026) 36 44 34 2
van Elden, Remelie	Prins Bernhardweg 18	6862 ZH	Oosterbeek	(026) 36 44 34 2
van Es, Xander	Haarlemweg 20	6843 AM	Arnhem	(026) 40 13 85 4
Fabels, Marc	Rijnkade 39c	6811 HA	Arnhem	(026) 4 90 4 008
van Ge en, Pim	Zwarteweg 3	6923 CK	Groessen	(0316) 7 27 9 39
Geurts, Karin	Mierlostraat 92	6844 DZ	Arnhem	(026) 41 16 30 0

Table 5: An example of vertical lay-out with elucidated horizontal lines

Top level TEX code:

```
\begin{fristimax}{AIO \today}{%
  size=A6,portrait,stack,%
  folds=1,fold=7mm,outervmar=4mm,%
  innervmar=0mm,outerhmar=2mm,
  corner=0mm,outline,%
  addresswidth=.425\hsize,%
  fontsize=12pt,stretch=.89,%
  textshape=\sffamily,%
  numbershape=\rmfamily\bfseries,%
  telarea=26,postarea=6}%
  \input ledenlst.dat
\end{fristimax}
```

AIO 26 september 1996   Netnrs. 26, tenzij anders vermeld. Postcodes beginnen met 6.
Woonplaatsen: (6)51–(6)54 Nijmegen   (6)721 Bennekom   (6)81–(6)84 Arnhem
(6)861–(6)862 Oosterbeek   (6)866 Heelsum   (6)871 Renkum   (6)874 Wolfheze
(6)88 Velp   (6)923 Groessen   (6)93 Westervoort   © FrisTi

Lieke Arts ————	**39 11 1 11**	*844* HC	*Gemertstraat 16*
Victoria ^{v}Asch^{v}Wijck	**24-66 16 9 57**	*601* CW	*Tunnelweg 4 Wijchen*
Sem Bezemer ———	**6 84 628 6**	*843* WR	*Alkmaarsingel 230*
Sarah Buddeke ——	**48 40 17 2**	*815* GM	*Sint Caeciliapad 35*
Hans ^{vdn}Bijlaard ——	**41 26 20 5**	*824* JN	*^{Ln} ^{v}Klarenbeek 105*
Quirijn ^{vdn}Bijlaard -	**318-1 21 75 6**	*721* MP	*W. Breukelderwg 15*
Arthur Constandse –	**9 48 92 99**	*823* DA	*Wielewaalstraat 5*
Arvid Ehlert ————	**7 85 05 67**	*844* HD	*Gemertstraat 17*
Ariette, Remelie ^{v}Elden	**36 44 34 2**	*862* ZH	*Pr Bernhardwg 18*
Xander van Es ———	**40 13 85 4**	*843* AM	*Haarlemweg 20*
Marc Fabels ————	**4 90 4 008**	*811* HA	*Rijnkade 39c Arnhem*
Pim van Ge en —	**316-7 27 9 39**	*923* CK	*Zwarteweg 3 Groess.*
Karin Geurts ————	**41 16 30 0**	*844* DZ	*Mierlostraat 92 Arnh.* *(e)*`loes.vd.tuin@tip.nl`
Thomas van Gurp —	**38 37 35 6**	*862* CG	*Annastraat 7 O'beek*
Herman Haverkort –	**35 16 7 23**	*814* CB	*Zijpendaalseweg 17* *(e)*`hh@fgbbs.iaf.nl`
Koen Haverkort ——	**51 39 36 1**	*823* AL	*Monnikensteeg 264*
Hester Hendriks ——	**41 49 20 7**	*862* CG	*Annastraat 27*
Ellen Hiemstra —	**317-9 24 25 5**	*866* ET	*Schutterspad 31*
Annebrecht 't Hoen -	**3 62 47 42**	*881* JN	*Schpsdr Overbeek 3*
Nanda van Huet ——	**3 25 47 68**	*852* MD	*Dullert 26 Huissen*
Tjeerd Kalsbeek ——	**3 61 471 3**	*826* JC	*V Kinsbergstr 46*

Table 6: Example of a fristimax list, based on the input of which some lines are presented in table 9 (only one side shown)

To conclude the section about stuffing techniques I will give an overview of some formatting utilities defined in hhfristi:

`\hhfrtel` gets four arguments: a one argument macro, the area code prefix, the area code postfix, and a phone number. The macro is used to typeset the whole: it is typically a font selection macro like `\textbf`. The prefix is typically "(0", the postfix ") ", and the phone number is specified by giving numbers, separated by dots. The first dot specifies the end of the area code, the following dots specify places where thin spaces should be inserted to group the digits. The area code, including prefix and postfix, is omitted if it equals the default area code defined by `\hhfr@telarea`.

`\hhfrlocalpostcode` gets three arguments: a macro to typeset the digits, a macro to typeset the alphabetic characters, and the postcode itself. In hhfristi a version for numeric postcodes and a version for Dutch postcodes are defined. The first digits or characters of the postcode are surrounded by `\default{` and `}` if they equal the default leading characters defined by `\hhfr@postarea`. By letting `\default \relax` or `\@gobble` one can choose if default characters should be typeset.

`\hhfrrecordpostcodes` gets three arguments. The third specifies a residence name, the first specifies its lowest postcode (or the lowest leading discriminating digits), the second its highest postcode. `\hhfrrecordpostcodes` inserts the residence name with its postcodes in the postcode list, unless it is already there.

`\hhfrpostcodes` gets two arguments: a title (for example: "Residences" and a macro to typeset postcodes. It typesets the postcode list filled by `\hhfrrecordpostcodes`.

`\hhfrgobblecentury` gets four arguments; a typical use is `\hhfrgobblecentury 19'\year`. If the expansion of `\year` starts with “19”, the first two digits are omitted and replaced by the apostrophe. `\year` will be fully typeset otherwise.

Data entry

I wanted to be able to specify the data to be listed in such a way that it would be independent of the lay-out. After all, I wanted to be able to use the same data file for very different lists, which do not only differ in the selection of data and the order in which the fields have to be presented, but also in the stuffing tricks used, and in the way in which the stuffing mechanism takes advantage of the structure of the data. For example: in some lists two persons living at the same address are put together on one line, in some lists they get two successive lines, and in other lists they would be typeset far apart from each other because the whole list would be sorted by first name instead of last name.

To facilitate all this I decided to enter the data in the following way. In principle all information about (for example) a person is given by listing pairs of tags and values. Tag and value are separated by a colon; pairs are separated by semicolons and spaces; the entire record is embraced. If multiple persons share the same information, the shared information is specified at the top level like described above. Then a list of persons is inserted instead of the remaining tag and value pairs. The list simply consists of a sequence of embraced subrecords. The same approach is taken if one person has multiple addresses, or telephone numbers, or whatever. Then the person’s name can be considered shared information for the phone numbers, and the phone numbers are put in a list. In table 7 examples are given of ‘singles’, persons sharing last name and address, persons sharing the address only, and even multiple persons, sharing the same address and sharing last names in groups.

The input shown in table 7 is part of the input for the credit card size horse owners and address list, of which both sides are shown in table 8, together with its TEX code. The hard work is done by the `fristidata` environment. `fristidata` gets two arguments: an initialisation sequence, and a list of patterns. The initialisation sequence, `\gdef\namen{}` in this example, is executed just before handling each data record. The list of patterns specifies what to do with each record. `fristidata` tries to match patterns and data from the input file. Whenever a match is found, the macros specified in the pattern are called and the next record is read from the file. For example: the first pattern matches data containing one value for the “prd” tag, one or no value for “str”, “hnr”, “pcd”, “wpl” and “tel”, one value for “fam”, and one or more values for “naam”. First, for each “naam” value, the macro `\voornaam` is called. Finally, the macro `\totaal` is called for handling the entire record.

If no pattern matches the data given, `fristidata` tries to restructure the data to fit a pattern. For example, the complex structure of names in the Goddijn-van-Weelden-family does not match any pattern. Therefore the structure of the data is simplified automatically by `fristidata` by ‘distributing’ the last names over the family members.

The macros which handle the actual processing of the data use some macros presented in the foregoing. A few are new:

`\hhfrqueue` gets three arguments: a macro name, a delimiter, and some text. It extends the definition of the specified macro by appending the text. If the macro was not empty the delimiter is used. For example, suppose that `\namen` has been defined by `\gdef\namen{}`. Then `\hhfrqueue\namen{ en }{Frans}` defines `\namen` to expand to “Frans”, and a following `\hhfrqueue\namen{ en }{Herman}` redefines `\namen` to “Frans en Herman”.

`\frat` gets one argument, a data tag. It expands to the value paired with the tag in the record concerned.

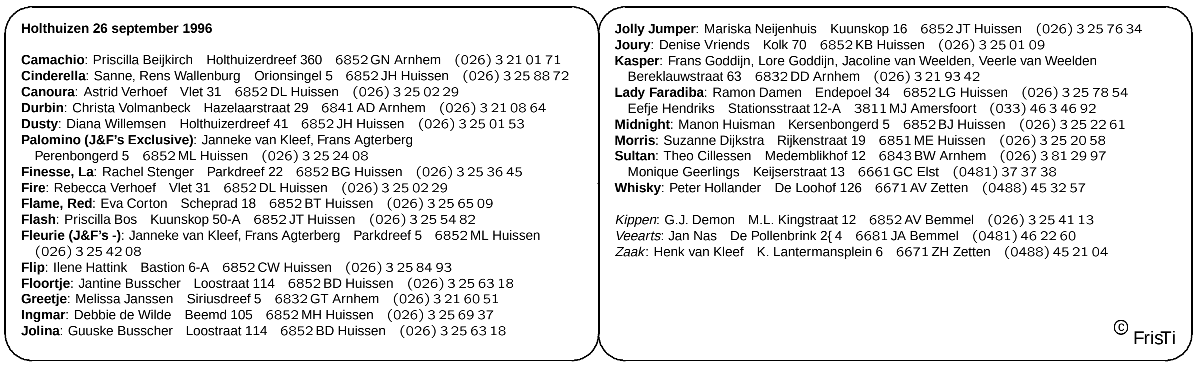
`\withfrat` gets two arguments, a data tag and some text. It expands to the text only if there is some value attached to the tag. For example: `\withfrat{hnr}{~\frat{hnr}}` is effective only if some value for “hnr” (the house number) is known, and if so, the number is typeset prefaced by a tie.

In table 9 you will find a few example lines from a data file which combines data tags and stuffing information. The stuffing information does not influence the typesetting of the data directly: it only indicates some stuffing *possibilities.* Wether or not these possibilities are used indeed is a matter of lay-out, which is not specified in the data file. The file regio.dat, which is input, defines macros with the names and postcodes of all cities and villages within about twenty kilometers from Arnhem. Street names often contain macros like `\weg` or `\straat`, which expand to stuffers which specify how to print those words in full and how to print them abbreviated.

```
{naam:Priscilla; fam:Beijkirch; tel:26.3.21.01.71;
  pcd:6852gn; str:Holthuizerdreef; hnr:360; wpl:Arnhem;
  prd:Camachio}%
{{naam:Sanne}{naam:Rens}; fam:Wallenburg; tel:26.3.25.88.72;
  pcd:6852jh; str:Orionsingel; hnr:5; wpl:Huissen;
  prd:Cinderella}%
{{naam:Janneke; fam:van Kleef}{naam:Frans; fam:Agterberg}; tel:26.3.25.24.08;
  pcd:6852ml; str:Perenbongerd; hnr:5; wpl:Huissen;
  prd:Palomino (J\&F’s Exclusive)}%
{{{naam:Frans}{naam:Lore}; fam:Goddijn}%
  {{naam:Jacoline}{naam:Veerle}; fam:van Weelden}; tel:26.3.21.93.42;
  pcd:6832dd; str:Bereklauwstraat; hnr:63; wpl:Arnhem;
  prd:Kasper}%
{{naam:Theo; fam:Cillessen; tel:26.3.81.29.97;
  pcd:6843bw; str:Medemblikhof; hnr:12; wpl:Arnhem}%
  {naam:Monique; fam:Geerlings; tel:481.37.37.38;
  pcd:6661gc; str:Keijserstraat; hnr:13; wpl:Elst};
  prd:Sultan}%
{naam:Jan; fam:Nas; tel:481.46.22.60;
  pcd:6681ja; str:De Pollenbrink; hnr:2--4; wpl:Bemmel;
  dienst:Veearts}%
```

Table 7: A part of the input for the card shown in table 8.

```
\def\voornaam{\hhfrqueue\namen{, }{\frat{naam}}}
\def\helenaam{\voornaam\hhfrqueue\namen~{\frat{fam}}}
\def\naamenadres{\hhfrqueue\namen{\hfill\protect\newline}{%
  \frat{naam}~\frat{fam}\quad\adres}}
\def\adres{\frat{str}~\frat{hnr}\nobreak\quad
  \protect\hhfrlocalpostcode{\textnr}{\uppercase}{\frat{pcd}}~\frat{wpl}\quad
  \withfrat{tel}{\protect\hhfrtel{\textnr}{(0}{)\,}{\frat{tel}}}}
\def\totaal{\raggedright\emergencystretch=.9\hsize\relax\hangindent1em
  \withfrat{prd}{\textbf{\frat{prd}}: }%
  \withfrat{dienst}{\textsl{\frat{dienst}}: }%
  \namen
  \withfrat{fam}{~\frat{fam}}%
  \withfrat{str}{\quad\adres}\par}
\begin{fristiform}{size=C,outline,seal=10pt,outervmar=6pt}
  \hhfrsizes\@vpt1\sffamily\parskip\z@\parindent\z@
  \noindent\hbox to \hsize{\textbf{Holthuizen \today}\hfil}
  \par\addvspace\baselineskip
  \begin{fristidata}{\gdef\namen{}}{%
    +prd ?str ?hnr ?pcd ?wpl ?tel +fam *1{+naam -> \voornaam} -> \totaal;
    +dienst ?str ?hnr ?pcd ?wpl ?tel +fam *1{+naam -> \voornaam} -> \totaal;
    +prd ?str ?hnr ?pcd ?wpl ?tel *2{+fam +naam -> \helenaam} -> \totaal;
    +prd *2{?str ?hnr ?pcd ?wpl ?tel +fam +naam -> \naamenadres} -> \totaal}
    \input{paarden.dat}
  \end{fristidata}
  \fristifoot
\end{fristiform}
```


Holthuizen 26 september 1996

Camachio: Priscilla Beijkirch Holthuizerdreef 360 6852 GN Arnhem (026) 3 21 01 71
Cinderella: Sanne, Rens Wallenburg Orionsingel 5 6852 JH Huissen (026) 3 25 88 72
Canoura: Astrid Verhoef Vlet 31 6852 DL Huissen (026) 3 25 02 29
Durbin: Christa Volmanbeck Hazelaarstraat 29 6841 AD Arnhem (026) 3 21 08 64
Dusty: Diana Willemsen Holthuizerdreef 41 6852 JH Huissen (026) 3 25 01 53
Palomino (J&F’s Exclusive): Janneke van Kleef, Frans Agterberg
Perenbongerd 5 6852 ML Huissen (026) 3 25 24 08
Finesse, La: Rachel Stenger Parkdreef 22 6852 BG Huissen (026) 3 25 36 45
Fire: Rebecca Verhoef Vlet 31 6852 DL Huissen (026) 3 25 02 29
Flame, Red: Eva Corton Scheprad 18 6852 BT Huissen (026) 3 25 65 09
Flash: Priscilla Bos Kuunskop 50-A 6852 JT Huissen (026) 3 25 54 82
Fleurie (J&F’s -): Janneke van Kleef, Frans Agterberg Parkdreef 5 6852 ML Huissen
(026) 3 25 42 08
Flip: Ilene Hattink Bastion 6-A 6852 CW Huissen (026) 3 25 84 93
Floortje: Jantine Busscher Loostraat 114 6852 BD Huissen (026) 3 25 63 18
Greetje: Melissa Janssen Siriusdreef 5 6832 GT Arnhem (026) 3 21 60 51
Ingmar: Debbie de Wilde Beemd 105 6852 MH Huissen (026) 3 25 69 37
Jolina: Guuske Busscher Loostraat 114 6852 BD Huissen (026) 3 25 63 18

Jolly Jumper: Mariska Neijenhuis Kuunskop 16 6852 JT Huissen (026) 3 25 76 34
Joury: Denise Vriends Kolk 70 6852 KB Huissen (026) 3 25 01 09
Kasper: Frans Goddijn, Lore Goddijn, Jacoline van Weelden, Veerle van Weelden
Bereklauwstraat 63 6832 DD Arnhem (026) 3 21 93 42
Lady Faradiba: Ramon Damen Endepoel 34 6852 LG Huissen (026) 3 25 78 54
Eefje Hendriks Stationsstraat 12-A 3811 MJ Amersfoort (033) 46 3 46 92
Midnight: Manon Huisman Kersenbongerd 5 6852 BJ Huissen (026) 3 25 22 61
Morris: Suzanne Dijkstra Rijkenstraat 19 6851 ME Huissen (026) 3 25 20 58
Sultan: Theo Cillessen Medemblikhof 12 6843 BW Arnhem (026) 3 81 29 97
Monique Geerlings Keijserstraat 13 6661 GC Elst (0481) 37 37 38
Whisky: Peter Hollander De Loohof 126 6671 AV Zetten (0488) 45 32 57

Kippen: G.J. Demon M.L. Kingstraat 12 6852 AV Bemmel (026) 3 25 41 13
Veearts: Jan Nas De Pollenbrink 2{ 4 6681 JA Bemmel (0481) 46 22 60
Zaak: Henk van Kleef K. Lantermansplein 6 6671 ZH Zetten (0488) 45 21 04

© FrisTi

Table 8: A credit card size list of horse owners and addresses, with the TEX code (two sides shown).

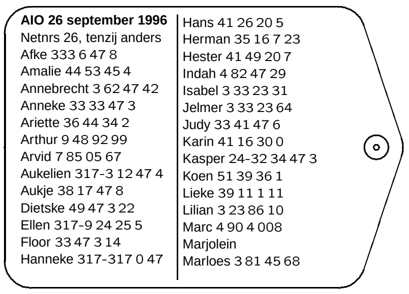
```
\input regio.dat
\def\Wijchen{\res{6601}{6605}{W"ych.}{W"ychen}}

\itm
{naam:Lieke; geb:15/3/1977; fam:Arts; tel:26.39.11.1.11;
  pcd:6844hc; str:Gemert\straat; hnr:16; wpl:\Arnhem}%
{naam:Victoria; geb:23/10/1978; fam:\van Asch\vvan Wijck; tel:24.66.16.9.57;
  fax:24.64.19.5.42; pcd:6601cw; str:Tunnel\weg; hnr:4; wpl:\Wijchen}%
{naam:Hans; geb:1/10; fam:\vdn Bijlaard; tel:26.41.26.20.5;
  pcd:6824jn; str:\kln{}{L\lng{aa}n v\lng{an}}{ }Klarenbeek; hnr:105; wpl:\Arnhem}%
{{naam:Ari\"ette; geb:1/9/1984}{naam:Remelie; geb:11/7/1986};
  fam:\van Elden; tel:26.36.44.34.2;
  pcd:6862zh; str:\kln{}{Pr\lng{ins}}{ }Bernhard\weg; hnr:18; wpl:\Oosterbeek}%
{naam:Xander; fam:\van Es; tel:26.40.13.85.4;
  pcd:6843am; str:Haarlem\weg; hnr:20; wpl:\Arnhem}%
```

Table 9: A few lines of the input file for the lists in table 6, 10 and 12 (data changed for privacy reasons).

Top level TEX code:

```
\begin{phonelist}{AIO \today}{%
  size=A9,eyepos=right,stack,columns=2,outline,%
  flush=left,fontsize=5pt,stretch=1.07,%
  textshape=\sffamily,numbershape=\nrfamily,%
  telarea=26}%
  \begin{fristisort}{naam}
    \input ledenlst.dat
  \end{fristisort}%
  \fristifoot[r]
\end{phonelist}
```

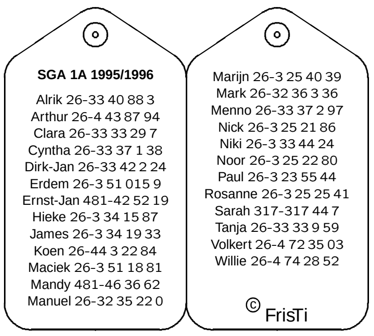


AIO 26 september 1996
Netnrs 26, tenzij anders
Afke 333 6 47 8
Amalie 44 53 45 4
Annebrecht 3 62 47 42
Anneke 33 33 47 3
Ariette 36 44 34 2
Arthur 9 48 92 99
Arvid 7 85 05 67
Aukelien 317-3 12 47 4
Aukje 38 17 47 8
Dietske 49 47 3 22
Ellen 317-9 24 25 5
Floor 33 47 3 14
Hanneke 317-317 0 47
Hans 41 26 20 5
Herman 35 16 7 23
Hester 41 49 20 7
Indah 4 82 47 29
Isabel 3 33 23 31
Jelmer 3 33 23 64
Judy 33 41 47 6
Karin 41 16 30 0
Kasper 24-32 34 47 3
Koen 51 39 36 1
Lieke 39 11 1 11
Lilian 3 23 86 10
Marc 4 90 4 008
Marjolein
Marloes 3 81 45 68

Table 10: Example of a phone list which can be carried on a key fob, based on the input of which some lines are presented in table 9 (only one side shown). We produce this list in a laminated version, with an eye in it. The font size used is five points, but choosing the fonts sensibly (e.g. cmssq for numbers) and using a decent printer bore its fruits: the conductor of the orchestra, who has to use all kinds of glasses to see properly, exclaimed: "even I can read it!". This phone list also demonstrates the use of the fristisort environment, which sorts the data by the field "naam" (name).

Top level TEX code:

```
\begin{phonelist}{SGA 1A 1995/1996}{%
  width=30mm,height=48.5mm,seal=3mm,%
  eyepos=top,outline,flush=centre,%
  fontsize=5pt,stretch=1.05,%
  textshape=\sffamily,numbershape=\nrfamily}%
  \addvspace{.5\baselineskip}%
  \begin{fristisort}{naam}
    \input sga1a.dat
  \end{fristisort}%
  \fristifoot[c]
\end{phonelist}
```

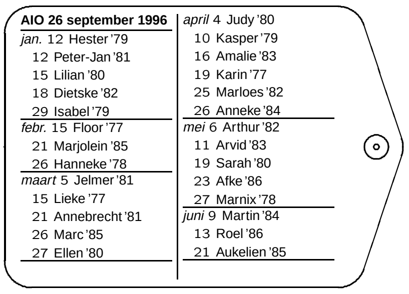


SGA 1A 1995/1996
Alrik 26-33 40 88 3
Arthur 26-4 43 87 94
Clara 26-33 33 29 7
Cyntha 26-33 37 1 38
Dirk-Jan 26-33 42 2 24
Erdem 26-3 51 015 9
Ernst-Jan 481-42 52 19
Hieke 26-3 34 15 87
James 26-3 34 19 33
Koen 26-44 3 22 84
Maciek 26-3 51 18 81
Mandy 481-46 36 62
Manuel 26-32 35 22 0
Marijn 26-3 25 40 39
Mark 26-32 36 3 36
Menno 26-33 37 2 97
Nick 26-3 25 21 86
Niki 26-3 33 44 24
Noor 26-3 25 22 80
Paul 26-3 23 55 44
Rosanne 26-3 25 25 41
Sarah 317-317 44 7
Tanja 26-33 33 9 59
Volkert 26-4 72 35 03
Willie 26-4 74 28 52
© FrisTi

Table 11: Another example of a key fob size phone list.

Top level TEX code:

```
\begin{cakelist}{AIO \today}{%
  size=A9,eyepos=right,stack,columns=2,outline,%
  fontsize=5pt,stretch=1.14,%
  textshape=\sffamily,numbershape=\nrfamily}%
  \input ledenlst.dat
\end{cakelist}
```



AIO 26 september 1996
jan. 12 Hester ’79
12 Peter-Jan ’81
15 Lilian ’80
18 Dietske ’82
29 Isabel ’79
febr. 15 Floor ’77
21 Marjolein ’85
26 Hanneke ’78
maart 5 Jelmer ’81
15 Lieke ’77
21 Annebrecht ’81
26 Marc ’85
27 Ellen ’80
april 4 Judy ’80
10 Kasper ’79
16 Amalie ’83
19 Karin ’77
25 Marloes ’82
26 Anneke ’84
mei 6 Arthur ’82
11 Arvid ’83
19 Sarah ’80
23 Afke ’86
27 Marnix ’78
juni 9 Martin ’84
13 Roel ’86
21 Aukelien ’85

Table 12: Example of a key fob size birthday calendar, based on the input of which some lines are presented in table 9 (only one side shown)