ACRO

V3.2  2020/05/10

Typeset Acronyms and other Abbreviations

Clemens Niederberger

https://github.com/cgnieder/acro/

contact@mychemistry.eu

ACRO allows you to define and manage acronyms and abbreviations. It can also be used for glossaries or nomenclatures.

Table of contents

I. Get started with ACRO
   1. Licence and requirements 3
   2. Glossary 3
   3. ACRO for the impatient 3

II. Comprehensive description of creation and usage of acronyms
   4. Setting options
      4.1. Load-time options .... 4
      4.2. Setup command .... 5
   5. Declaring acronyms and other abbreviations
      5.1. Basic properties .... 6
      5.2. Properties related to plural and indefinite forms .... 7
      5.3. Properties related to formatting 8
      5.4. Properties related to the created PDF file .... 9
      5.5. Further properties .... 10
   6. Using acronyms .... 10
   7. Alternative short forms .... 12
   8. The first or full appearance .... 12
   9. Single appearances of an acronym .... 13

10. Printing the list
   10.1. The main command and its options .... 14
   10.2. Add page numbers to the list .... 16
   10.3. Filter lists using tags .... 16
   10.4. Local lists .... 17

11. Formatting .... 18

12. Plural forms and other endings .... 20
   12.1. The plural ending and the plural form .... 20
   12.2. Other endings .... 21

13. Articles .... 21
   13.1. Indefinite forms .... 21
   13.2. Other articles .... 22

14. Foreign language acronyms .... 22
15. Uppercasing 24

16. Citing and indexing 25
   16.1. Citing 25
   16.2. Indexing 26

17. Barriers 27

18. Trailing tokens 28
   18.1. What is it about? 28
   18.2. How does it work? 29

19. Using or resetting acronyms 30

20. Bookmarks, backlinks and accessibility support 31
   20.1. Backlinks 31
   20.2. Bookmarks 31
   20.3. PDF comments 31
   20.4. Accessibility support 32

21. Localisation 33

III. Extending \texttt{ACRO} 34

22. Background 34
   22.1. Templates 34
   22.2. Objects 35

23. Templates 35
   23.1. Pre-defined templates 35
      23.1.1. Acronym templates 35
      23.1.2. List templates 36
      23.1.3. Header templates 37
   23.2. Defining new templates 37

23.3. Commands to be used in template definitions 38
   23.3.1. Commands for common uses 38
   23.3.2. Commands for usage in acronym templates 39
   23.3.3. Commands for usage in list templates 39
   23.4. New acronym templates 40
   23.5. New list templates 41
   23.6. New heading templates 43

24. Endings 43

25. Articles 44

26. Translations 45

27. Properties 46

28. Own acronym commands 46
   28.1. Background 46
   28.2. Create commands for possessive endings 48

29. Own \texttt{ACRO} style files 49

IV. Appendix 50

A. Examples 50
B. Acronyms 54
C. References 55
D. Index 55
Part I.
Get started with **ACRO**

1. Licence and requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the \LaTeX\ Project Public License (\LPPL), version 1.3c or later (http://www.latex-project.org/\lppl.txt). The software has the status “maintained.”

2. Glossary

**articles** Articles are prefixes to acronyms, usually separated with a blank. Different types of articles are mutually exclusive.

**endings** Endings are postfixes to acronyms, usually not separated from the acronym. Different types of endings are mutually exclusive.

**load-time option** A load-time option is a package option of \ACRO{} which must be set as option to \usepackage\{\ACRO{}\}.

**option** An option is a package option of \ACRO{} which must set with \acsetup\. It cannot be set as option to \usepackage. Options usually also can be set in the optional argument of \ac{} and friends.

**property** A property is an option to the second argument of the \DeclareAcronym{} command. They are options of an individual acronym if you will.

**template** A template determines how different objects of \ACRO{} are printed. This includes the acronyms themselves but also for example the list of acronyms as a whole.

**translations** Localisation strings which can be modified.

3. **ACRO** for the impatient

Acronyms are defined in the preamble via the command

\DeclareAcronym\{\textit{id}\}\{}\{\textit{properties}\}\}

where \textit{id} is a unique string to identify the acronym and \textit{properties} is a key/value list of acronym properties. These include:

\begin{itemize}
\item **short** = \{\textit{text}\}\} \hspace{1cm} (required)
\begin{itemize}
\item The short form of the acronym. \textit{This property is required:} an acronym must have a short form.
\end{itemize}
\item **long** = \{\textit{text}\}\} \hspace{1cm} (required)
\begin{itemize}
\item The long form of the acronym. \textit{This property is required:} an acronym must have a description.
\end{itemize}
\end{itemize}
In its simplest form an acronym needs a short and a long form. Please note that both properties must be set.

In the document acronyms are used with these commands:

\ac{⟨id⟩} \Ac{⟨id⟩}
\ac prints the acronym ⟨id⟩, the first time with full description and every subsequent use only the abbreviated form. \Ac does the same but uppercases the first letter – this may be needed at the beginning of a sentence.

\acs{⟨id⟩} \Acs{⟨id⟩}
\acs prints the short form of the acronym ⟨id⟩. \Acs does the same but uppercases the first letter.

\acl{⟨id⟩} \Acl{⟨id⟩}
\acl prints the long form of the acronym ⟨id⟩. \Acl does the same but uppercases the first letter.

\acf{⟨id⟩} \Acf{⟨id⟩}
\acf prints the full form of the acronym ⟨id⟩. \Acf does the same but uppercases the first letter.

Let’s say you defined CD as follows:

\begin{tabular}{ll}
  first & \ac{cd} \\
  second & \ac{cd} \\
  long & \acl{cd} \\
  short & \acs{cd} \\
  full & \acf{cd} \\
\end{tabular}

Then the usage is

4. Setting options

4.1. Load-time options

ACRO knows only a small set of load-time options which can be used as argument to \usepackage:
4. Setting options

version = 2|3
Initial: 3
The option allows you to use the last version prior to the update to version 3. This may help if you don’t have the time to fix issues after upgrading to the new version.

upgrade = true|false
Initial: true
When this option is used ACRO tries to give as much helpful and meaningful warning or error messages when a deprecated or removed command, property, or option is used. This is especially useful if you are upgrading from version 2. The option will initially be true for a few months after the upgrade to version 3.

4.2. Setup command

All options of ACRO that have not been mentioned in section 4.1 have to be set up either with this command

\acsetup\{\langle options\rangle\}

or as option to other commands. If the latter is possible then it is described when the corresponding commands are explained. Options usually follow a key/value syntax and are always described in the following way:

option
An option without a value. Those options are very rare if there are any.

option = \{\langle value\rangle\}
Initial: preset
An option where a value can be given. The pre-set value is given to the right.

option = choiceA|choiceB|choiceC
Initial: choiceB
An option with a determined set of choices. The underlined value is chosen if the option is given without value.

option = true|false
A boolean option with only the choices true and false.

module\option
An option at a deeper level belonging to the module module.

All of the above is probably clear from an example (using real options):

\begin{verbatim}
\acsetup{
    make-links = true , % boolean
    index , % boolean
    format = \emph , % standard
    list / local , % boolean option of the list module
    list/display = all % choice option of the list module
}
\end{verbatim}
Part II.
Comprehensive description of creation and usage of acronyms

5. Declaring acronyms and other abbreviations

All acronyms have to be declared in the preamble with the following command in order to be used in the document. Any usage of an acronym which has not been declared leads to an error message.

\DeclareAcronym{⟨id⟩}{⟨list of properties⟩}

The basic command for declaring an acronym where ⟨id⟩ is a unique string identifying the acronym. Per default this is case sensitive which means id is different from ID, for example.

The command understands a number of properties which are listed in the following sections. This is a comprehensive overview over the existing properties. Many properties are also explained in more detail in later sections of this manual.

case-sensitive = true|false
Initial: false
When this is set you can write the ID of the acronym upper- or lower- or mixed case and it is recognized by ACRO as the same. This might be useful when the acronym appears in the page header, for example.

In its simplest form an acronym needs a short and a long form. Please note that both properties must be set.

5.1. Basic properties

short = {⟨text⟩} (required)
The short form of the acronym. This property is required: an acronym must have a short form.

Maybe you mostly have simple acronyms where the ID and short form are the same. In that case you can use

use-id-as-short = true|false
Initial: false
to use the ID of the acronym as short form. For more complicated cases this would still allow you to set the short form.

long = {⟨text⟩} (required)
The long form of the acronym. This property is required: an acronym must have a description.

alt = {⟨text⟩} (initially empty)
Alternative short form.
5. Declaring acronyms and other abbreviations

extra = \{(\text{text})\} \hspace{1cm} \text{(initially empty)}

Extra information to be added in the list of acronyms.

foreign = \{(\text{long form in foreign language})\} \hspace{1cm} \text{(initially empty)}

Can be useful when dealing with acronyms in foreign languages, see section 14 on page 22 for details.

long-post = \{(\text{text})\} \hspace{1cm} \text{(initially empty)}

\text{(text)} is appended to the long form of the acronym in the text but not in the list of acronyms.

post = \{(\text{text})\} \hspace{1cm} \text{(initially empty)}

\text{(text)} is appended to the acronym in the text but not in the list of acronyms.

single = \{(\text{text})\} \hspace{1cm} \text{if unused then equal to long}

If provided \text{(text)} will be used instead of the long form if the acronym is only used a single time and the option single has been set, see section 9 on page 13.

sort = \{(\text{text})\} \hspace{1cm} \text{if unused then equal to short}

If used the acronym will be sorted according to this property instead of its ID.

tag = \{(\text{csv list})\} \hspace{1cm} \text{(initially empty)}

The tag(s) of an acronym.

cite = \{(\text{prenote})\}|\{(\text{postnote})\}|\{(\text{citation keys})\} \hspace{1cm} \text{(initially empty)}

A citation that is printed to the acronym according to an option explained later.

before-citation = \{(\text{text})\} \hspace{1cm} \text{(initially empty)}

\text{(text)} is prepended to the citation of the acronym when and where the citation is printed.

index = \{(\text{text})\} \hspace{1cm} \text{(initially empty)}

This property allows to overwrite the automatic index entry with an arbitrary one. See section 16.2 on page 26 for details.

index-sort = \{(\text{text})\} \hspace{1cm} \text{if unused then equal to sort}

If you use the option index every occurrence of an acronym is recorded to the index and sorted by its short form or (if set) by the value of the sort property. This property allows to set an individual sorting option for the index. See section 16.2 on page 26 for details.

5.2. Properties related to plural and indefinite forms

short-plural = \{(\text{text})\} \hspace{1cm} \text{Initial: s}

The plural ending appended to the short form.

short-plural-form = \{(\text{text})\} \hspace{1cm} \text{(initially empty)}

The plural short form of the acronym; replaces the short form when used instead of appending the plural ending.

long-plural = \{(\text{text})\} \hspace{1cm} \text{Initial: s}

The plural ending appended to the long form.
5. Declaring acronyms and other abbreviations

long-plural-form = {{text}}

Plural long form of the acronym; replaces the long form when used instead of appending the plural ending.

alt-plural = {{text}}

The plural ending appended to the alternative form.

alt-plural-form = {{text}}

The plural alternative form of the acronym; replaces the alternative form when used instead of appending the plural ending.

foreign-plural = {{text}}

The plural ending appended to the foreign form.

foreign-plural-form = {{text}}

Plural foreign form of the acronym; replaces the foreign form when used instead of appending the plural ending.

short-indefinite = {{text}}

Indefinite article for the short form.

long-indefinite = {{text}}

Indefinite article for the long form.

alt-indefinite = {{text}}

Indefinite article for the alternative form.

5.3. Properties related to formatting

format = {{code}}

The format used for both short and long form of the acronym.

short-format = {{code}}

if unused then equal to format

The format used for the short form of the acronym.

long-format = {{code}}

if unused then equal to format

The format used for the long form of the acronym.

first-long-format = {{code}}

if unused then equal to long-format

The format used for the first appearance of the long form of the acronym.

alt-format = {{code}}

if unused then equal to short-format

The format used for the alternative form of the acronym. If this is not given the short format will be used.

extra-format = {{code}}

(initially empty)

The format used for the additional information of the acronym.
5. Declaring acronyms and other abbreviations

foreign-format = \{⟨code⟩\} (initially empty)
      The format used for the foreign form of the acronym.

single-format = \{⟨code⟩\} if unused then equal to long-format
      The format used for the acronym if the acronym is only used a single time.

list-format = \{⟨code⟩\} if unused then equal to long-format
      The format used for the long form of the acronym in the list if the list template supports it. All
      pre-defined list templates do support it.

first-style = long-short|short-long|short|long|footnote (initially empty)
      The style of the first appearance of the acronym, see also section 8 on page 12.

single-style = long-short|short-long|short|long|footnote (initially empty)
      The style of a single appearance of the acronym, see also section 9 on page 13.

5.4. Properties related to the created PDF file

pdfstring = \{⟨pdfstring⟩\} if unused then equal to short
      Used as PDF string replacement in bookmarks when used together with the hyperref [ORT20]
      or the bookmark package [Obe19].

pdfcomment = \{⟨text⟩\}
      Sets a tooltip description for an acronym. For actually getting tooltips you also need an
      appropriate setting of the options pdfcomment/cmd and pdfcomment/use, see also section 20.3
      on page 31.

short-acc = \{⟨text⟩\} if unused then equal to short
      Sets the ActualText property as presented by the accsupp package for the short form of the
      acronym.

long-acc = \{⟨text⟩\} if unused then equal to long
      Sets the ActualText property as presented by the accsupp package for the long form of the
      acronym.

alt-acc = \{⟨text⟩\} if unused then equal to alt
      Sets the ActualText property as presented by the accsupp package for the alternative short
      form of the acronym.

foreign-acc = \{⟨text⟩\} if unused then equal to foreign
      Sets the ActualText property as presented by the accsupp package for the foreign form of the
      acronym.

extra-acc = \{⟨text⟩\} if unused then equal to extra
      Sets the ActualText property as presented by the accsupp package for the extra information of
      the acronym.
sets the ActualText property as presented by the accsupp package for a single appearance of the acronym.

\texttt{list-acc} = \{\langle \text{text} \rangle \}
if unused then equal to \texttt{list}

Sets the ActualText property as presented by the accsupp package for the appearance in the list of acronyms.

\section{Further properties}

\texttt{list} = \{\langle \text{text} \rangle \}
if unused then equal to \texttt{long}

If specified this will be written in the list as description instead of the long form if the corresponding list template supports it.

\texttt{foreign-babel} = \{\langle \text{language} \rangle \}
(initially empty)

The babel \cite{Bra19} or polyglossia \cite{Cha19} language of the foreign form. This language is used to wrap the entry with \texttt{\foreignlanguage{\langle \text{language} \rangle}} if either babel or polyglossia is loaded. You’ll need to take care that the corresponding language is loaded by babel or polyglossia.

\texttt{foreign-locale} = \{\langle \text{language} \rangle \}
(initially empty)

The language name that is output when the option \texttt{locale/display} is used. If this property is not set then the appropriate value might be derived from \texttt{foreign-babel}. See section \ref{sec:locale} on page 22 for details.

\section{Using acronyms}

There are a number of commands to use acronyms with. Their names always follow the same pattern which should make their usage intuitive immediately.

All of these commands have a starred form which means “don’t count this as usage”. All of these commands also have an optional argument that allows to set options for that usage only.

\texttt{\acrocommand*{\langle \text{options} \rangle \{\langle \text{id} \rangle \}}}

This is the general syntax of all of the commands listed below. The star and the optional argument is left away for the sake of readability. \textit{A command \texttt{\acrocommand} does not actually exist.}

\texttt{\ac{\langle \text{id} \rangle} \ Ac{\langle \text{id} \rangle} \ acp{\langle \text{id} \rangle} \ Acp{\langle \text{id} \rangle} \ iac{\langle \text{id} \rangle} \ Iac{\langle \text{id} \rangle}}

\texttt{ac} prints the acronym \texttt{\langle \text{id} \rangle}, the first time with full description and every subsequent use only the abbreviated form. \texttt{Ac} does the same but uppercases the first letter – this may be needed at the beginning of a sentence. The commands \texttt{acp} and \texttt{Acp}, resp., print the corresponding plural forms. The commands \texttt{iac} and \texttt{Iac}, resp., print indefinite forms.

\texttt{\acs{\langle \text{id} \rangle} \ Acs{\langle \text{id} \rangle} \ acsp{\langle \text{id} \rangle} \ Acs{\langle \text{id} \rangle} \ iacs{\langle \text{id} \rangle} \ Iacs{\langle \text{id} \rangle}}

\texttt{acs} prints the short form of the acronym \texttt{\langle \text{id} \rangle}. \texttt{Acs} does the same but uppercases the first letter. The commands \texttt{acsp} and \texttt{Acs}, resp., print the corresponding plural forms. The commands \texttt{iacs} and \texttt{Iacs}, resp., print indefinite forms.
6. Using acronyms

\acl{\langle id \rangle} \Acl{\langle id \rangle} \aclp{\langle id \rangle} \Aclp{\langle id \rangle} \iacl{\langle id \rangle} \Iacl{\langle id \rangle}
\acl\ prints the long form of the acronym \langle id \rangle. \Acl\ does the same but uppercases the first letter. The commands \aclp\ and \Aclp, resp., print the corresponding plural forms. The commands \iacl\ and \Iacl, resp., print indefinite forms.

\aca{\langle id \rangle} \Aca{\langle id \rangle} \acap{\langle id \rangle} \Acap{\langle id \rangle} \iac{\langle id \rangle} \Iac{\langle id \rangle}
\aca\ prints the alternative short form of the acronym \langle id \rangle. \Aca\ does the same but uppercases the first letter. The commands \acap\ and \Acap, resp., print the corresponding plural forms. The commands \iac\ and \Iac, resp., print indefinite forms.

\acf{\langle id \rangle} \Acf{\langle id \rangle} \acfp{\langle id \rangle} \Acfp{\langle id \rangle} \iacf{\langle id \rangle} \Iacf{\langle id \rangle}
\acf\ prints the full form of the acronym \langle id \rangle. \Acf\ does the same but uppercases the first letter. The commands \acfp\ and \Acfp, resp., print the corresponding plural forms. The commands \iacf\ and \Iacf, resp., print indefinite forms.

The usage should be clear. Let’s assume you have defined an acronym UFO like this:

```latex
\DeclareAcronym{ufo}{
  short = UFO,
  long = unidentified flying object,
  foreign = unbekanntes Flugobjekt,
  foreign-plural-form = unbekannte Flugobjekte,
  foreign-babel = ngerman,
  long-indefinite = an
}
```

Then typical outputs look like this:

```
\acl{ufo}     // unidentified flying object (unbekanntes Flugobjekt, UFO)
\iacl{ufo}    // a UFO
\iac{ufo}     // an unidentified flying object
\iac{ufo}     // An unidentified flying object (unbekanntes Flugobjekt, UFO)
\iacf{ufo}    // unidentified flying objects (unbekannte Flugobjekte, UFOs)
```

In a number of contexts all acronym commands act as if their starred form is used: in the table of contents, in the list of figures, and in the list of tables. The same is true for floats and the measuring phase of common table environments like `tabularx` or `\texttable`.
7. Alternative short forms

Sometimes expressions have two different short forms. An example might be JPEG which also often is JPG. This is what the property \texttt{alt} is there for.

\texttt{alt} = \{\langle\textit{text}\rangle\}

Alternative short form.

Let’s define JPEG:

```latex
\DeclareAcronym{jpg}{
  short = JPEG ,
  sort = jpeg ,
  alt = JPG ,
  long = Joint Photographic Experts Group
}
```

And let's see how to use it:

```latex
\ac{jpg} \\
\ac{jpg} \\
\aca{jpg}
```

Joint Photographic Experts Group (JPEG or JPG)
JPEG
JPG

As you can see the full form shows both short forms of the acronym. This could be changed by altering the template for the full form, see section 23 on page 35 and section 8. The alternative form is also printed in the list of acronyms, see section A on page 54. This can also be changed by altering the template for the list, again see section 23.

8. The first or full appearance

If an acronym is used for the first time with \texttt{\ac} (after any number of usages with the starred forms of the usage commands listed in section 6 on page 10) or if an acronym is used \texttt{\acf}, then the first or full appearance of the acronym is printed.\footnote{This usually requires at least two \LaTeX{} runs until it is stable.}

The first or full appearance of an acronym is determined by this option:

\texttt{first-style} = \texttt{long-short|short-long|short|long|footnote} \hspace{1cm} Initial: \texttt{long-short}

The style of the first appearance of the acronym. This options sets the appearance for all acronyms. Available options in reality are the names of all defined templates of the type \texttt{acronym}. All pre-defined templates can be found in section 23.1 on page 35.
9. Single appearances of an acronym

It might be desirable to set the first appearance of an acronym individually. This is possible by setting the corresponding property:

\texttt{first-style = long-short|short-long|short|long|footnote} \hspace{1em} (initially empty)

The style of the first appearance of the acronym.

Let's again look at an example:

\begin{tabular}{ll}
1. & \texttt{\textbackslash acf[first-style=long-short]{cd} \textbackslash} \\
2. & \texttt{CD (compact disc)} \\
3. & \texttt{compact disc (CD)} \\
4. & \texttt{CD} \\
5. & \texttt{compact disc} \\
\end{tabular}

This also demonstrates the use of the optional argument. An example of an abbreviation that should have \texttt{long} as first appearance might be “\texttt{etc.}”, defined like this

\begin{tabular}{l}
1. \texttt{\textbackslash DeclareAcronym{etc}} \{} \\
2. \texttt{short = etc} \texttt{\textbackslash acdot} \texttt{,} \\
3. \texttt{long = et cetera ,} \\
4. \texttt{format = \textbackslash textit} \texttt{,} \\
5. \texttt{first-style = long ,} \\
6. \texttt{plural =} \\
7. \texttt{\}}
\end{tabular}

and output like this:

\begin{tabular}{l}
1. \texttt{\textbackslash ac{etc}}, \texttt{\textbackslash ac{etc}} \texttt{\textbackslash ac{etc}}. \\
2. \texttt{et cetera, etc. etc.}
\end{tabular}

The command \texttt{\textbackslash acdot} is explained in section 18 on page 28. Basically it checks if a dot follows and outputs a dot if not.

\textbf{9. Single appearances of an acronym}

If an acronym is used only once (not counting usages with the starred forms of the usage commands listed in section 6 on page 10), then the single appearance of the acronym is printed.\footnote{This usually requires at least two \LaTeX \ runs until it is stable.}

The single appearance of an acronym is determined by this option:

\texttt{single = true|false|\langle number \rangle} \hspace{1em} Initial: \texttt{false}

This option determines wether a single appearance of an acronym counts as \textit{usage}. It might be
In such cases that an acronym is simply printed as long form and not added to the list of acronym. This is what this option does. With \langle\text{number}\rangle the minimal number of usages can be given that needs to be exceeded. \texttt{single = \{1\}} is the same as \texttt{single = \{true\}}.

\texttt{single-style = long-short|short-long|short|long|footnote} \quad \text{Initial: long}

The style of the single appearance of an acronym. Can be used to determine how a single appearance is printed if the option \texttt{single} has been set. This option sets the appearance for all acronyms. Available options in reality are the names of all defined templates of the type \texttt{acronym}. All pre-defined templates can be found in section 23.1 on page 35.

If you like you can also set the single appearance of an acronym individually:

\texttt{single = \{}\langle\text{text}\rangle\}\text{ if unused then equal to long}

If provided \langle\text{text}\rangle will be used instead of the long form if the acronym is only used a single time and the option \texttt{single} has been set.

\texttt{single-format = \{}\langle\text{code}\rangle\}\text{ if unused then equal to long-format}

The format used for the acronym if the acronym is only used a single time.

\texttt{single-style = long-short|short-long|short|long|footnote} \quad \text{(initially empty)}

The style of the single appearance of the acronym.

Let’s again look at an example. The acronym PNG is defined as follows:

\begin{verbatim}
\DeclareAcronym{png}{
  short = PNG ,
  long = Portable Network Graphics ,
  first-style = short-long ,
  single-style = short
}
\end{verbatim}

And it is used only once in this manual:\footnote{You will find it in the list of acronyms in section A nonetheless as this document does list/display = \{all\}.}

\begin{verbatim}
\ac{png} PNG
\end{verbatim}

Please be aware that \texttt{\ac{f}} would still print the full form, of course.

10. Printing the list

10.1. The main command and its options

The main idea is simple: just place
\printacronyms[(options)]

where you want the list to appear. It may require several (most times two) \LaTeX runs for it to stabilize so look out for any warnings from \acro requiring to re-run.

The options controlling the list are these:

\begin{itemize}
  \item \texttt{list/template} = description|table|longtable|lof|toc \hspace{1cm} Initial: description
    Choose the template to create the list with. See more on this in sections 23 on page 35 and A on page 50.
  \item \texttt{list/sort} = \texttt{true}|false \hspace{1cm} Initial: \texttt{true}
    Decide whether to sort the list of acronyms alphabetically or to print it in order of definition.
  \item \texttt{list/display} = \texttt{all}|\texttt{used} \hspace{1cm} Initial: \texttt{used}
    Decide whether to print only the acronyms actually used in the document or all acronyms which have been declared in the preamble.
  \item \texttt{list/exclude} = \{⟨csv list of tags⟩\} \hspace{1cm} (initially empty)
    Set a list of tags to exclude from the list. Only acronyms not belonging to one of these tags will be included.
  \item \texttt{list/include} = \{⟨csv list of tags⟩\} \hspace{1cm} (initially empty)
    Set a list of tags to include in the list. Only acronyms belonging to one of these tags will be included.
  \item \texttt{list/heading} = none|section|section*|chapter|chapter* \hspace{1cm}
    Choose the heading template for the list of acronyms.
    This only has an effect if the list template supports it. All pre-defined templates \texttt{do} support it.
  \item \texttt{list/name} = \{(⟨text⟩\}
    Overwrites the text which is used in the heading.
    This only has an effect if the list template supports heading templates \texttt{and} the heading templates support it. All pre-defined heading templates \texttt{do} support this.
  \item \texttt{list/preamble} = \{(⟨text⟩\}
    Set a preamble to be placed between heading and actual list.
    This only has an effect if the list template supports it. All pre-defined templates \texttt{do} support it.
  \item \texttt{list/locale/display} = \texttt{true}|\texttt{false} \hspace{1cm} Initial: \texttt{false}
    This option determines whether the language of the foreign form is printed or not.
    This only has an effect if the list template supports foreign forms. All pre-defined templates \texttt{do} support them.
\end{itemize}

All these options can be set with \acsetup globally or locally as options to \printacronyms. In the latter case omit the leading \texttt{list}:
10. Printing the list

\acsetup{list/display=all,list/exclude=units}

or

\printacronyms{display=all,exclude=units}

10.2. Add page numbers to the list

If you want to include the page numbers where the acronyms have been used in the list of acronym you can use these options:

**pages/display** = first|all|none

Decide whether to include page numbers in the list of acronyms and whether to add the first page or every page. When you choose first and have hyperref loaded you will also get a backlink to that page.

**pages/seq/use** = true|false

Initial: true

Turns a two-page range into \( \text{num} \) f. (sequens) and a three-page range into \( \text{num} \) ff. (sequentestes) when set to true.

**pages/seq/pre** = \{\text{code}\}

Initial: \,

\text{code} is inserted between the page number and the sequens or sequentes symbol.

**pages/seq/threshold** = \{\text{num}\}

Initial: 3

The threshold for a page range to be turned into sequentestes. A page range above the threshold is still typeset as a range:

\( \langle \text{num}_1 \rangle - \langle \text{num}_2 \rangle \).

**pages/fill** = \{\text{code}\}

Initial: \dotfill

This is the code that is placed between acronym description and actual page numbers.

**pages/name** = true|false

Initial: false

If set to true the page numbers are preceded with p. or pp.

10.3. Filter lists using tags

With the property tag you can assign one or more tags to an acronym. These tags can be used to filter the list of acronyms.

**tag** = \{csv list\}

The tag(s) of an acronym.

**list/exclude** = \{csv list of tags\}

Set a list of tags to exclude from the list. Only acronyms not belonging to one of these tags will be included.

**list/include** = \{csv list of tags\}

Set a list of tags to include in the list. Only acronyms belonging to one of these tags will be included.
10. Printing the list

Let’s look at an example. This manual declares these two acronyms with the tag *city*:

```latex
\begin{verbatim}
\DeclareAcronym{la}{
  short = LA ,
  long = Los Angeles,
  plural = ,
  tag = city
}
\DeclareAcronym{ny}{
  short = NY ,
  long = New York ,
  plural = ,
  tag = city
}
\end{verbatim}
```

We can now use this to either print a list *without* these acronyms by saying

```latex
\printacronyms[exclude=city]
```

or print a list *with only* these acronyms with

```latex
\printacronyms[include=city,heading=none]
```

| LA | Los Angeles |
| NY | New York    |

If you use both `exclude` and `include` and list a tag in both `exclude` takes precedence over `include`.

```latex
\printacronyms[exclude={a,b},include={b,c}]
```

would only print acronyms with tag `c`.

### 10.4. Local lists

Maybe you like a list of acronyms for each chapter in a book which only lists the acronyms used within this chapter. You need to do three things: set

```latex
\texttt{barriers/use = true|false}  \hspace{1cm} \text{Initial: false}
```

this option to `true`, place
Please don’t use page numbers together with local lists for the time being. If an acronym appears in more than one list both lists would contain the same page numbers instead of only the ones local to barriers. For the similar reasons please also don’t use make-links together with local lists. This might be resolved on day.
11. Formatting

format = \{⟨code⟩\}  
(initially empty)

The format used for both short and long form of the acronym.

short-format = \{⟨code⟩\}  
if unused then equal to format

The format used for the short form of the acronym.

long-format = \{⟨code⟩\}  
if unused then equal to format

The format used for the long form of the acronym.

first-long-format = \{⟨code⟩\}  
if unused then equal to long-format

The format used for the first appearance of the long form of the acronym.

alt-format = \{⟨code⟩\}  
if unused then equal to short-format

The format used for the alternative form of the acronym. If this is not given the short format will be used.

extra-format = \{⟨code⟩\}  
(initially empty)

The format used for the additional information of the acronym.

foreign-format = \{⟨code⟩\}  
(initially empty)

The format used for the foreign form of the acronym.

single-format = \{⟨code⟩\}  
if unused then equal to long-format

The format used for the acronym if the acronym is only used a single time.

list-format = \{⟨code⟩\}  
if unused then equal to long-format

The format used for the long form of the acronym in the list if the list template supports it. All pre-defined list templates do support it.

first-style = long-short|short-long|short|long|footnote  
(initially empty)

The style of the first appearance of the acronym, see also section 8 on page 12.

single-style = long-short|short-long|short|long|footnote  
(initially empty)

The style of a single appearance of the acronym, see also section 9 on page 13.

Per default the individual formatting instructions are additive to the global ones. This can be changed through the option

format/replace = true|false  
Initial: false

With this option active local options will replace the global ones.

Let's see an example:

\begin{verbatim}
\DeclareAcronym{pdf}{
  short = pdf ,
  long = Portable Document Format ,
  short-format = \textsc{shape}
}
\end{verbatim}
12. Plural forms and other endings

12.1. The plural ending and the plural form

Not in all languages plural forms are as easy as always appending an “s”. Not even English. Sometimes there’s other endings instead. This is why ACRONYM has quite a number of different properties related to plural forms or endings:

\[
\begin{align*}
\text{short-plural} & = \{(text)\} \\
\text{short-plural-form} & = \{(text)\} \\
\text{long-plural} & = \{(text)\} \\
\text{long-plural-form} & = \{(text)\} \\
\text{alt-plural} & = \{(text)\} \\
\text{alt-plural-form} & = \{(text)\} \\
\text{foreign-plural} & = \{(text)\} \\
\text{foreign-plural-form} & = \{(text)\}
\end{align*}
\]

Initial: $s$

- The plural ending appended to the short form.
- The plural short form of the acronym; replaces the short form when used instead of appending the plural ending.
- The plural ending appended to the long form.
- Plural long form of the acronym; replaces the long form when used instead of appending the plural ending.
- The plural ending appended to the alternative form.
- The plural alternative form of the acronym; replaces the alternative form when used instead of appending the plural ending.
- The plural ending appended to the foreign form.
- Plural foreign form of the acronym; replaces the foreign form when used instead of appending the plural ending.

German is full of such examples.

---

4. German is full of such examples.
13. Articles

There are two options which allow to change the default values for the whole document:

\[ \text{short-plural-ending} = \{ \langle \text{text} \rangle \} \]

Initial: \text{s}

Defines the plural ending for the short forms to be \langle \text{text} \rangle.

\[ \text{long-plural-ending} = \{ \langle \text{text} \rangle \} \]

Initial: \text{s}

Defines the plural ending for the long forms to be \langle \text{text} \rangle.

Now let’s see two simple examples demonstrating the two different kinds of plural settings:

\begin{verbatim}
\SetAcronym{sw}{}{SW}{Sammelwerk}{e}
\SetAcronym{MP}{}{MP}{Member of Parliament}{Members of Parliament}
\end{verbatim}

The first one has another plural ending than the usual “s”. The second one has a different plural form altogether because appending an “s” would give a wrong form:

\begin{verbatim}
\acfp{sw} \par
\acfp{MP}
\end{verbatim}

\text{Sammelwerk (SWs)} \text{Members of Parliament (MPs)}

12.2. Other endings

Besides plural endings there are other ones like the genitive case, for example. This is why \texttt{ACRO} generalized the concept. Section 24 on page 43 explains in detail how to define and use additional endings.

13. Articles

13.1. Indefinite forms

Indefinite forms can be a problem if the short and the long form of acronyms have different indefinite articles.\footnote{This may very well be a language specific issue.}

…

\text{This may very well be a language specific issue.}
14. Foreign language acronyms

And what good would it be to use a package like ACRO if you have to keep track of first and second uses, anyway? This is why UFO should be defined like we did on page 11. We then can just use the dedicated commands and let them decide for us:

\begin{tabular}{ll}
\texttt{\acs{ufo}} & a unidentified flying object (unbekanntes Flugobjekt, UFO) \\
\texttt{\iac{ufo}} & an UFO
\end{tabular}

The commands which also output the indefinite article all start with an “i” and have all been described in section 6 on page 10 already: \texttt{\iac, \Iac, \ic, \Ic, \iacf, \Iacf}.

13.2. Other articles

There might be cases – most likely depending on your language – when you would like to have other articles behaving similar to the indefinite ones. Section 25 explains in detail how to define and use additional articles.

14. Foreign language acronyms

Sometimes and in some fields more often than in others abbreviations are used that are derived from another language. ACRO provides a number of properties for such cases:

\begin{itemize}
  \item foreign = \{⟨long form in foreign language⟩\} (initially empty)
  \begin{itemize}
    \item Can be useful when dealing with acronyms in foreign languages, see section 14 for details.
  \end{itemize}
  \item foreign-plural = \{⟨text⟩\} Initial: s
  \item foreign-plural-form = \{⟨text⟩\} (initially empty)
  \item foreign-format = \{⟨code⟩\} (initially empty)
  \item foreign-babel = \{⟨language⟩\} (initially empty)
\end{itemize}

The plural ending appended to the foreign form.

Plural foreign form of the acronym; replaces the foreign form when used instead of appending the plural ending.

The format used for the foreign form of the acronym.

The babel or polyglossia language of the foreign form. This language is used to wrap the entry with \texttt{\foreignlanguage{⟨language⟩}} if either babel or polyglossia is loaded. You’ll need to take care that the corresponding language is loaded by babel or polyglossia.
14. Foreign language acronyms

\texttt{foreign-locale = \{(language)\}}

(initially empty)

The language name that is output when the option \texttt{locale/display} is used. If this property is not set then the appropriate value might be derived from \texttt{foreign-babel}.

There are also some options:

\texttt{foreign/display = true|false}

Determine whether to hide or display the foreign form.

\texttt{list/foreign/display = true|false}

\texttt{Initial: true}

\texttt{New}

Determine whether to hide or display the foreign form in the list of acronyms.

\texttt{locale/display = true|false}

This option determines whether the language of the foreign form is printed or not when the full form of the acronym is printed.

\texttt{list/locale/display = true|false}

The same but for the list of acronyms.

\texttt{locale/format = \{(code)\}}

Determine how said language is formatted when printed. The last command in \texttt{(code)} may take a mandatory argument.

Let’s say you are writing a German document and are using the abbreviation ECU for Steuergerät which stems from the English “Electronic Control Unit”. Then you can define it as follows:

\begin{verbatim}
\DeclareAcronym{ecu}{
  short = ECU ,
  long = Steuergerät ,
  foreign = Electronic Control Unit ,
  foreign-babel = english ,
  foreign-locale = englisch
}
\end{verbatim}

Now the abbreviation is introduced so that everyone understands the confusion:

\begin{verbatim}
\ac{ecu} \par
\acsetup{locale/display,locale/format=\emph}
\acf{ecu}
\end{verbatim}

Steuergerät (Electronic Control Unit, ECU)
Steuergerät (\textit{englisch}: Electronic Control Unit, ECU)

The property \texttt{foreign-babel} is used for ensuring correct hyphenation as long as you use babel or polyglossia and load the corresponding language, too. If you are writing your document in English then \texttt{ACRO} is able to deduce the language used for the \texttt{“locale”} field by itself:
15. Uppercasing

Depending on the kind of abbreviations you have and depending on their definition and maybe also depending on your language the long and sometimes also the short forms need to start with an uppercase letter at the beginning of a sentence while it starts with a lowercase letter otherwise.

For this `acro` provides uppercase versions for all predefined acronym commands listed in section 6. The usage is self-explaining:

```
\DeclareAcronym{ufo}{
  short = UFO ,
  long = Unidentified Flying Object
}
```

```
\acsetup{locale/display,first-style=short-long}
\acf{ufo}
```

```
There was \iacl{ufo} hovering \dots \par
\Aclp{ufo} were hovering \dots
```

There was an unidentified flying object hovering …
Unidentified flying objects were hovering …

If you defined them with uppercase letters to begin with then these commands have no effect, of course.

```
\DeclareAcronym{eg}{
  short = e.g\acdot ,
  long = for example ,
  foreign = exempli gratia ,
  foreign-babel = latin ,
  short-format = \textit ,
  foreign-format = \textit
}
```

```
\Acsetup{locale/display,first-style=short-long}
\Acf{eg}
```

```
e.g. (Latin: exempli gratia: for example)
```
There are a number of options to control the uppercasing behavior:

**uppercase/first**
The default setting. Converts the first letter to uppercase.

**uppercase/title**
This is just a synonym of `first`.

**uppercase/all**
Converts all letters to uppercase.

**uppercase/none**
Converts all letters to lowercase.

**uppercase/cmd = \{\textit{command}\}**
All of the above options just choose the right command using this option internally. This means you can choose a different behavior altogether by setting this option to something else. For example you could use `\textit{capitalisewords}` from the package `mfirstuc` [Tal17]. The command needs to have one mandatory argument.

There may be reasons to exclude short forms from being uppercased. This can be controlled by this option:

**uppercase/short = \textit{true}|false}**
Initial: true

It allows you to disable the mechanism for the `short` and `alt` properties.

## 16. Citing and indexing

### 16.1. Citing

Acronyms can be given cite keys. This makes it possible to add a citation reference automatically when the acronym is used for the first time.

Let’s see an example first. NY has been defined like this:

```latex
\DeclareAcronym{ny}{
  short = NY ,
  long = New York ,
  plural = ,
  tag = city ,
  cite = NewYork
}
```

The property `cite` will now trigger `ACRO` to input `\cite{NewYork}` after the acronym:

```latex
\ac{ny} \hspace{1cm} \text{New York (NY) [Wik20]}
```
Depending on the citation style (and probably other factors, too) it might be desirable to add the citation rather inside the parentheses together with the short form of the acronym and even cited with a different command. For cases like these ACRO offers a number of options:

\texttt{cite/cmd} = \{\texttt{citation command}\} \hspace{1cm} \text{Initial:} \ \texttt{\cite}

Choose the command with which citations are printed.

\texttt{cite/group} = \texttt{true}|\texttt{false} \hspace{1cm} \text{Initial:} \ \texttt{false}

Decide whether to group citations with the short form in the parentheses. The template must support this. ACRO’s pre-defined templates do support it.

\texttt{cite/display} = \texttt{first}|\texttt{all}|\texttt{none} \hspace{1cm} \text{Initial:} \ \texttt{first}

Decide whether to output the citation in the first/full usage only or always or never.

\texttt{cite/pre} = \{\texttt{text}\} \hspace{1cm} \text{Initial:} \ \texttt{\nobreakspace}

Arbitrary code directly output before the citation.

\texttt{cite/group/cmd} = \{\texttt{citation command}\} \hspace{1cm} \text{Initial:} \ \texttt{\cite}

Choose the command with which grouped citations are printed.

\texttt{cite/group/pre} = \{\texttt{text}\} \hspace{1cm} \text{Initial:} \ \text{, _}

Arbitrary code directly output before the citation in the grouped case.

If for example you use biblatex’s authoryear style [LKW19] you might want to have settings like these:

\begin{verbatim}
\acsetup{
    cite/group = true ,
    cite/cmd = \parencite ,
    cite/group/cmd = \cite
}
\end{verbatim}

\begin{verbatim}
\acsetup{cite/display = all}
\acf{ny} \ \\
\ac{ny}
\end{verbatim}

New York (NY, Wikipedia 2020)
NY (Wikipedia 2020)

\section*{16.2. Indexing}

Maybe you want to add your acronyms to an index. In that case it is probably desirable to let ACRO make this automatically. In the simplest case just enable it:

\begin{verbatim}
\acsetup{index/enable = true}
\end{verbatim}
17. Barriers

index/use = true|false  
Initial: false
Enable indexing.

index/cmd = \{\{index command\}\}
Choose a command for indexing.  
Initial: \index

index/disable = \{\{code\}\}
Sometimes it is desirable to change the meaning of a command inside an index entry. For the  
entries created by \acro this can be achieved with this option.

index/clear
This option clears the disable list.
While these options set global behavior there are also properties to set them for an acronym individually.

index = \{\{text\}\}
This property allows to overwrite the automatic index entry with an arbitrary one.

index-sort = \{\{text\}\}
If you use the option index every occurrence of an acronym is recorded to the index and sorted 
by its short form or (if set) by the value of the sort property. This property allows to set an 
individual sorting option for the index.

no-index = true|false  
Initial: true
This property allows to exclude an acronym from being indexed.

This manual is an example for the indexing feature. Each acronym from section A on page 54 
that has been used in this manual is also listed in the index.

17. Barriers

The main purpose of the concept of barriers is to be able to have local lists of acronyms. This concept does a little bit more than that, though, which should become clear from the following options:

barriers/use = true|false  
Initial: false
Activate usage of barriers. Otherwise the command \acbarrier just does nothing except writing a warning in the log.

barriers/reset = true|false  
Initial: false
When set to true the acronym usage is reset for all acronyms at a barrier. The first use of \ac after a barrier will again look like the \acf.

barriers/single = true|false  
Initial: false
When set to true a single usage of an acronym between two barriers with \ac will look according 
to the chosen style as explained in section 9 on page 13. This option only has an effect when 
the option single is used as well.
There are two natural barriers in a document: \begin{document} and \end{document}. You can add an arbitrary number of additional barriers with \acbarrier. For this command to have any effect you must set barriers/use to true!

It usually takes two or even three \LaTeX runs until acronym usages between barriers are properly counted.

## 18. Trailing tokens

### 18.1. What is it about?

\acro has the possibility to look ahead for certain tokens and switch a boolean variable if it finds them. Per default \acro knows about three tokens: the “dot” (.), the “dash” (-) and the “babel-hyphen” (\textit{\textbackslash babelhyphen}).

You have seen an example for this already:

```latex
\DeclareAcronym{etc}{
  short = etc\acdot,
  long = et cetera,
  format = \textit,
  first-style = long,
  plural =
}
```

The macro \acdot recognizes if a dot is directly following. It only prints a dot if it doesn’t.

```latex
\ac{etc} and \ac{etc}.
```

Another example: let’s say you’re a German scientist, you have

```latex
\DeclareAcronym{PU}{
  short = PU,
  long = Polyurethan,
  long-plural = e
}
```

and you use it the first time like this:
then according to German orthography and typesetting rules this should be printed as

“Polyurethan(\text{PU})-Hartschaum”

i. e., with no space between long and short form.

This works because the template \text{long-short} \textsuperscript{6} uses \texttt{acspace} at the appropriate place and the manual setup does

\texttt{\acssetup{activate-trailing-tokens = dash}}

\texttt{acspace} looks ahead for a trailing dash and adds a space it it doesn’t find it.

18.2. How does it work?

Tokens to look for can be defined and activated through the following options:

\texttt{\acssetup{trailing/define = \{token\}\{\langle name\rangle\}}}

Defines token \langle name\rangle and tells ACRO look for \langle token\rangle if \langle name\rangle is activated.

\texttt{\acssetup{trailing/activate = \{\langle csv list of token names\rangle\}}}

Tell ACRO to look for trailing tokens. This is done by giving a csv list of the internal names of the tokens. Per default only dot is activated.

\texttt{\acssetup{trailing/deactivate = \{\langle csv list of token names\rangle\}}}

Tell ACRO not to look for trailing tokens. This is done by giving a csv list of the internal names of the tokens.

The package itself does this:

\begin{verbatim}
\acssetup{
  trailing/define   = . (dot) ,
  trailing/define   = , (comma) ,
  trailing/define   = - (dash)
  trailing/define   = \{babelhyphen\} (babel-hyphen) ,
  trailing/activate = (dot, comma)
}
\end{verbatim}

\footnote{6. The template that is used by default for the first appearance.}
19. Using or resetting acronyms

In order to make use of this mechanism there is the following command:

\aciftrailing{\{csv list of token names\}}{\{true\}}{\{false\}}

Check if one of the tokens listed in \{csv list of token names\} is following and either place \{true\} or \{false\} in the input stream.

This command is used to define the two commands you already know:

\acdot
Inserts \abbrdot if no dot follows.

\acspace
Inserts a \space if no dash or babel-hyphen follows.

\abbrdot
Inserts .\@

The definitions are equivalent to the following code:

\begin{verbatim}
1. \newcommand*=\acdot{\aciftrailing{dot}}{{\abbrdot}}
2. \newcommand*=\acspace{\aciftrailing{dash,babel-hyphen}}{{\space}}
\end{verbatim}

You are of course free to redefine them according to your needs.

19. Using or resetting acronyms

Sometimes it is necessary to mark an acronym as used before it actually has been used or to mark an acronym as unused even though it has been used. You have already seen one of the commands which make it possible:

\acuse{\{csv list of acronym ids\}}
Every acronym given in the list will be marked as used.

\acuseall
Every acronym is marked as used.

\acreset{\{csv list of acronym ids\}}
Every acronym given in the list will be reset.

\acresetall
Every acronym will be reset.

In a number of contexts all acronym commands act as if their starred form is used: in the table of contents, in the list of figures, and in the list of tables. The same is true for floats and the measuring phase of common table environments like tabularx or \ltxtable.
20. Bookmarks, backlinks and accessibility support

20.1. Backlinks

When `acro` is used together with the package `hyperref` then you can make use of the following option:

```
make-links = true|false  Initial: false
```

If this is activated then every short or alternative appearance of an acronym will be linked to its description in the list of acronyms.

This will fail miserably together with local lists if an acronym appears in more than one list. This might be resolved on day.

20.2. Bookmarks

Since bookmarks (which are created by the `hyperref` or the `bookmark` packages) can only contain simple text `acro` simplifies the output of the acronym commands when they appear in a bookmark. Although the output can be modified with a dedicated template-mechanism there is no user interface at the moment. Contact me at https://github.com/cgnieder/acro/issues if you need it.

Acronyms have the property `pdfstring`:

```
pdfstring = \{⟨pdfstring⟩\}  if unused then equal to short
```

Used as PDF string replacement for the short form in bookmarks when used together with the `hyperref` or the `bookmark` package.

This is for acronyms like

```latex
\DeclareAcronym{pdf}{
  short = pdf ,
  long = Portable Document Format ,
  short-format = \textscshape ,
  pdfstring = PDF
}
```

where the bookmark would write “pdf” instead of “PDF” if the property were not set.

20.3. PDF comments

Some people like see comments in the PDF when they’re hovering with the mouse over the short form of an acronym. This can be achieved.

```
pdfcomments/use = true|false  Initial: false
```

This enables the creation of PDF comments.
20. Bookmarks, backlinks and accessibility support

\pdfcomment{cmd} = \{\{code\}\}

Initial: \pdftooltip{#1}{#2}

Chooses the command for actually creating the comment. You must refer to the printed output in the PDF with #1 and to the comment with #2. The default command \pdftooltip is provided by the package pdfcomment [Kle18]. You must load it in order to use it.

Only acronyms where the corresponding property has been set will get comments:

\pdfcomment{pdfcomment} = \{(text)\}

Sets a tooltip description for an acronym.

20.4. Accessibility support

ACRO supports the accsupp package [Obe18] when you also load hyperref. Then ACRO uses

```
\begin{accsupp}{method = pdfstringdef, ActualText = \{PDF\}}
\textsc{pdf}
\end{accsupp}
```

for an acronym defined like this:

```
\DeclareAcronym{pdf}{
  short = pdf,
  long = Portable Document Format,
  short-format = \scshape,
  pdfstring = PDF,
  short-acc = PDF
}
```

Without accessibility support when a string like “PDF” is copied from the PDF and pasted you get “pdf”. If you don’t care about that simply don’t load accsupp and ignore this section.

You have a few options to be able to manipulate what ACRO does here but I recommend to stay with the default settings:

\acsupp/use = \{true|false\}

Initial: true

When this is true and the package accsupp is loaded then accessibility support is used.

\acsupp/options = \{(text)\}

(Initially empty)

Additional option to be passed to \begin{accsupp}. See the accsupp manual for possible settings.

\acsupp/method = \{(method)\}

Initial: pdfstringdef

The method used by \begin{accsupp}. See the accsupp manual for possible values.

The “ActualText” that is used by ACRO always defaults to the values of the acronym properties themselves. You can choose these values individually by setting the corresponding properties:
21. Localisation

There are places when \acro uses text strings which depend on the language of the document. In order to recognize the language from babel of polyglossia and print the strings in the correct language \acro uses the translations [Nie20].
If the language is detected incorrectly or you want \acro{} to use another language than it detects you can use the following option:

\texttt{language = auto\langle language\rangle}

Initial: \texttt{auto}

The default setting \texttt{auto} lets \acro{} detect the language setting automatically. Valid choices are all language names known to the package translations. Mostly just type your language and it should work.

\acro{} only provides support for a handful of languages. You can easily teach \acro{} your language – see section 26 on page 45 – if it isn’t supported, yet.⁷

\texttt{\acrotranslate\langle key\rangle}

This command fetches the translation of \langle key\rangle for the current language. It is meant for usage in template definitions.

Available keywords and their English, French, and German translations are shown in table 1.

### Part III.

### Extending \acro{}

#### 22. Background

##### 22.1. Templates

One of the core ideas of \acro{} version 3.0 is the use of templates which manage how different how anything is printed, from the output of \texttt{\ac} and friends to the list of acronyms. \acro{} uses three types of templates:

---

⁷ If you like you can always open an issue at https://github.com/cgnieder/acro/issues and provide your translations so I can add them to \acro{.}
23. **Templates**

**acronym** These templates can be used to define *acronym commands*, see section 28 on page 46.

**list** These templates are used by the `\printacronyms` command.

**heading** These templates only make sense if a *list* template uses `\acroheading`. This command makes use of them.

How these templates are defined, which are available from the start and how they are used is explained in section 23.

### 22.2. Objects

**ACRO** uses certain kinds of objects in some of its commands. It is possible to define own such objects:

**articles** Per default only the “indefinite” article is defined. But it is possible to define and add other articles to **ACRO**. This is explained in section 13.2 on page 22.

**endings** Per default only the ending “plural” is defined. But it is possible to define and add other endings to **ACRO**. This is explained in section 12.2 on page 21.

**properties** You have already learned about properties. It is possible to define and add further acronym properties to **ACRO**. This is explained in section 27 on page 46.

**translations** **ACRO** uses localisation strings at a number of places. It is possible to change these strings and add further strings. This is explained in section 21 on page 33.

### 23. Templates

#### 23.1. Pre-defined templates

##### 23.1.1. Acronym templates

**alt**

Display the alternative form of an acronym.

**first**

This is a *pseudo* template which always displays what is set through the option `first-style` or the property `first-style`.

**footnote**

A template for the first appearance where the long form is printed in a footnote.

**long**

Display the long form of an acronym.

**long-short**

A template for the first appearance where the long form is printed and the short form follows in parentheses.
23. Templates

**first**
This is a *pseudo* template which always displays what is set through the `single-style` or the property `single-style`.

**short**
Display the short form of an acronym.

**short-long**
A template for the first appearance where the short form is printed and the long form follows in parentheses.

**show**
A template which writes all properties of an acronym into the log file.

### 23.1.2. List templates

**description**
The default list style which places the short form in the item of a `description` environment and adds the all the rest as description of the item.

**lof**
A style which mimicks the list of figures. This style does not support page ranges.

**longtable**
A style that uses a `longtable` environment for building the list. This needs the `longtable` package [Car19] loaded.

**longtabu**
A style that uses a `longtabu` environment for building the list. This needs the `longtable` package and the `tabu` package [Che19] loaded.\(^8\)

**supertabular**
A style that uses a `supertabular` environment for building the list. This needs the `supertabular` package [BJ20] loaded.

**tabular**
A style that uses a `tabular` environment for building the list. Since a `tabular` cannot break across pages this is only suited for short lists.

**toc**
A style which mimicks the table of contents. This style does not support page ranges.

---

\(^8\) Please note that this package currently is un-maintained and has a number of open bugs. For further information refer to https://github.com/tabu-issues-for-future-maintainer/tabu
23. Templates

23.1.3. Header templates

addchap
Only defined in a KOMA-Script class and if \chapter is defined. Uses \addchap for the heading.

addsec
Only defined in a KOMA-Script class. Uses \addsec for the heading.

chapter
Only defined if \chapter is defined. Uses \chapter for the heading.

chapter*
Uses \chapter* for the heading.

none
Displays nothing.

section
Uses \section for the heading.

section*
Uses \section* for the heading.

23.2. Defining new templates

For the definition of templates these commands are available:

\NewAcroTemplate[⟨type⟩]{⟨name⟩}{⟨code⟩}
This defines a template of type ⟨type⟩ with the name ⟨name⟩ which inserts ⟨code⟩ when used. A template of type ⟨type⟩ with name ⟨name⟩ must not exist. The default type is acronym.

\RenewAcroTemplate[⟨type⟩]{⟨name⟩}{⟨code⟩}
This re-defines a template of type ⟨type⟩ with the name ⟨name⟩ which inserts ⟨code⟩ when used. A template of type ⟨type⟩ with name ⟨name⟩ must exist. The default type is acronym.

\SetupAcroTemplate[⟨type⟩]{⟨name⟩}{⟨code⟩}
☆ New
Adds ⟨code⟩ to the beginning of the template ⟨name⟩ of type ⟨type⟩. The default type is acronym.

\SetupNextAcroTemplate[⟨type⟩]{⟨name⟩}{⟨code⟩}
☆ New
Adds ⟨code⟩ to the beginning of the next use of the template ⟨name⟩ of type ⟨type⟩. The default type is acronym.

* \AcroTemplateType
Within a template this expands to the ⟨type⟩ of the current template.

* \AcroTemplateName
Within a template this expands to the ⟨name⟩ of the current template.

How to use these commands is best explained by examples of how the existing templates have been defined. The following sections will show several examples for their usage.
23. Templates

23.3. Commands to be used in template definitions

\acro provides and uses a large number of commands that are meant to be used in template definitions and that often are useless or will raise errors if used outside. Depending on their purpose the commands can be used in different types of templates or only in certain types of templates.

In the descriptions below a \* indicates a fully expandable command when used in an `edef`, `write` or in `expanded`.

A `TF` always refers to a `true` and `false` branch and indicates that three commands exist: one exactly as described, one with only the `T` and the `true` branch, and one with only the `F` and the `false` branch. So `\acroifTF` means there is `\acroifTF`, `\acroifT`, and `\acroifF`, where `\acroifT` and `\acroifF` each have an argument less than `\acroifTF`.

23.3.1. Commands for common uses

\* `\acrolistname`

Expands to whatever is currently set with `list/name`.

\acrowrite{⟨property⟩}

Prints the property `⟨property⟩` of the current acronym. Depending on the circumstances this prints the property together with an article or an ending either in uppercase or lowercase form. Default is the lowercase form without ending or article. The actual outcome is determined by switches which are explained in section 28 on page 46.

\acroformat{⟨type⟩}{⟨text⟩}

This formats `⟨text⟩` according to `⟨type⟩` where `⟨type⟩` has either been set as property or as option from the `format` module. Valid values are `short`, `long`, `alt`, `extra`, `foreign`, `list`, and `first-long`.

\acroshow{⟨property⟩}

For debugging purposes: writes the property `⟨property⟩` of the current acronym to the log file.

\acroifTF{⟨property⟩}{⟨true⟩}{⟨false⟩}

Checks if the property `⟨property⟩` has been set for the current acronym and either leaves `⟨true⟩` or `⟨false⟩` in the input stream.

\acroifbooleanTF{⟨property⟩}{⟨true⟩}{⟨false⟩}

Returns `⟨true⟩` if the boolean property `⟨property⟩` has been set to `true` and `⟨false⟩` otherwise.

\acroifallTF{⟨properties⟩}{⟨true⟩}{⟨false⟩}

Checks if all properties in the csv list `⟨properties⟩` have been set for the current acronym and either leaves `⟨true⟩` or `⟨false⟩` in the input stream.

\acroifanyTF{⟨properties⟩}{⟨true⟩}{⟨false⟩}

Checks if any of the properties in the csv list `⟨properties⟩` has been set for the current acronym and either leaves `⟨true⟩` or `⟨false⟩` in the input stream.
\acroiftagTF{⟨tag⟩}{⟨true⟩}{⟨false⟩}
Checks if the current acronym has been given the tag ⟨tag⟩ and either leaves ⟨true⟩ or ⟨false⟩ in the input stream.

\AcroPropertiesMap{⟨code⟩}
Maps over all defined acronym properties. Within ⟨code⟩ you can refer to the current property with #1.

23.3.2. Commands for usage in acronym templates

\acroifusedTF{⟨true⟩}{⟨false⟩}
Checks if the current acronym has been used before and either leaves ⟨true⟩ or ⟨false⟩ in the input stream.

\acroiffirstTF{⟨true⟩}{⟨false⟩}
Checks if the current usage of the current acronym is the first time and either leaves ⟨true⟩ or ⟨false⟩ in the input stream.

\acroifsingleTF{⟨true⟩}{⟨false⟩}
Checks if the current acronym is used a single time and either leaves ⟨true⟩ or ⟨false⟩ in the input stream.

\acrogroupcite

23.3.3. Commands for usage in list templates

\acroifchapterTF{⟨true⟩}{⟨false⟩}
This just check if \chapter is defined. Used in the toc template.

\acroifpagesTF{⟨true⟩}{⟨false⟩}
This is ⟨true⟩ if the option pages/display is set, and the current acronym is not single, and has at least one page number. ⟨false⟩ otherwise.

\acropages{⟨first⟩}{⟨range⟩}
If \acroifpagesTF would be ⟨false⟩ this would do nothing. Otherwise, if pages/display is first it prints the first page number, preceded by ⟨first⟩ if pages/name is true. If pages/display is all it prints the page range, preceded by ⟨range⟩ if pages/name is true.

\acronopagerange
This disables page ranges. Used in the toc and lof templates.

\acropagefill
If \acroifpagesTF would be ⟨false⟩ this would do nothing. Otherwise it prints whatever is set by pages/fill.
\textbf{23. \textit{Templates}}

\texttt{\textbackslash acronymsmap\{(\texttt{code})\}}

Maps over the acronyms in order of appearance in the list. Which acronyms these are depends on settings. They might only have certain tags, be ones local to barriers, …

Within \texttt{\{\tt code\}\#1} refers to the current ID of the acronym. Also \texttt{\textbackslash AcronymID} expands to the current ID. The latter is important for all the commands that check or print properties of acronyms.

\texttt{\textbackslash acronymsmapTF\{(\texttt{code})\}\{\{\texttt{true}\}\}\{\{\texttt{false}\}\}}

This does the same as \texttt{\textbackslash acronymsmap} and also leaves \texttt{\{\tt true\}} in the input stream if the list is not empty and \texttt{\{\tt false\}} otherwise. This is useful to trigger a rerun warning.

\texttt{\textbackslash AcronymTable}

This is an empty token list at the beginning of a list template.

\texttt{\textbackslash AcroAddRow\{(\texttt{code})\}}

Adds \texttt{\{\tt code\}} to the right of \texttt{\textbackslash AcronymTable} and ensures that \texttt{\textbackslash AcronymID} has the correct global definition for this code. With this the code for the \textit{tabular} template and other table templates can be built in a comfortable way.

\texttt{\textbackslash AcroNeedPackage\{(\texttt{package})\}}

Checks if the package \texttt{\{\tt package\}} is loaded and throws an error otherwise.

\texttt{\textbackslash AcroRerun}

Triggers \texttt{ACRO} to throw an “empty list” rerun warning.

\textbf{23.4. \textit{New acronym templates}}

Some templates are quite short and self-explaining:

\begin{verbatim}
\NewAcroTemplate{short}{\acrowrite{short}}
\end{verbatim}

Some are a little bit more elaborate:

\begin{verbatim}
\NewAcroTemplate{alt}{%
  \acroifTF{alt}{%
    {\acrowrite{alt}}%
    {\acrowrite{short}}%
  }%
}\end{verbatim}

And some templates need to do a lot more:

\begin{verbatim}
\NewAcroTemplate{long-short}{%
  \acroiffirstTF{%
    \acrowrite{long}%
  }%
}\end{verbatim}
23. Templates

23.5. New list templates

This section shows the definition of three templates: description, tabular, and toc.

First the description template:

```latex
\NewAcroTemplate[list]{description}{% 
\acroheading 
\acropreamble 
\begin{description} 
\acroifT{alt}{\acrowrite{alt}}% 
\item\acroifT{foreign,extra}{(}% 
\acroifT{foreign}{\acrowrite{foreign}\acroifT{extra}{, }}% 
\acroifT{extra}{\acrowrite{extra}}% 
\acroifanyT{foreign,extra}{)}% 
\acropagefill 
\acropages 
\acrotranslate{page}\nobreakspace} 
\acrotranslate{pages}\nobreakspace}% 
\item\AcroRerun 
\end{description} 
}% 
\end{description} 
```

The following shows how to define templates using some kind of table environment. Special care is necessary due to the way \LaTeX\ tables work: first the table body is built and only then the table itself is printed:

```latex
\NewAcroTemplate[list]{tabular}{% 
\AcroNeedPackage{array}% 
\acronymssmapF{% 
\AcroAddRow{% 
\acrowrite{short}% 
\acroifanyT{foreign,extra}{}% 
\acroifT{foreign}{\acrowrite{foreign}\acroifT{extra}{, }}% 
\acroifT{extra}{\acrowrite{extra}}}% 
\acroifanyT{foreign,extra}{)}% 
\acropagefill 
\acropages 
\acrotranslate{page}\nobreakspace} 
\acrotranslate{pages}\nobreakspace}% 
\item\AcroRerun 
\} 
```

41
23. Templates

\acroifT{alt}{/acrowrite{alt}}
&
\acrowrite{list}%
\acroifanyT{foreign,extra}{()}
\acroifT{foreign}{/acrowrite{foreign}\acroifT{extra}{, }}%
\acroifT{extra}{/acrowrite{extra}}%
\acroifanyT{foreign,extra}{}
\acropagesfill
\acropages
\{/acrotranslate{page}\nobreakspace
\{/acrotranslate{pages}\nobreakspace%
\tabularnewline
\}
}\%
\{/AcroRerun\%
\acroheading
\acropreamble
\par
\noindent
\begin{tabular}{>{\bfseries}lp{.7\linewidth}}
\AcronymTable
\end{tabular}
\}

\NewAcroTemplate[list]{toc}{%
\acroheading
\acropreamble
\acronopagerange
\acronymssmapF{
\contentsline{\acroifchapterTF{chapter}{section}}
{\acrowrite{short}\acroifT{alt}{/acrowrite{alt}}}{}
{}%
\contentsline{\acroifchapterF{sub}section}
{\acroifT{foreign}{/acrowrite{foreign}\acroifT{extra}{, }}%
\acroifT{extra}{/acrowrite{extra}}%
\acroifanyT{foreign,extra}{}%}
\acroifanyT{foreign,extra}{}%}
\acroifanyT{foreign,extra}{}%
\acroifanyT{foreign,extra}{}%
{\acropages{}{}}%
{}%
{\AcroRerun}
\}

42
23.6. New heading templates

24. Endings

Referring to section 12.2 on page 21 this section explains how to define and use additional endings.

\DeclareAcroEnding{⟨name⟩}{⟨short default⟩}{⟨long default⟩}

This command can be used to define properties and options analogous to the plural endings which have been defined this way:

\DeclareAcroEnding{plural}{s}{s}

In general \DeclareAcroEnding{⟨foo⟩}{⟨x⟩}{⟨y⟩} defines these options

\acro{⟨foo⟩}

This command tells the template that the ending ⟨foo⟩ should be used.

In addition another command is defined which is meant to be used in template definitions.

\acro{⟨foo⟩}

This command tells the template that the ending ⟨foo⟩ should be used.


Section 28 on page 46 has an example of how this can be used to define a possessive ending and commands that make use of them like this:

\begin{quote}
\acro{MP}

Member’s of Parliament (MP’s)
\end{quote}

\section{Articles}

Referring to section 13.2 on page 22 this section explains how to define and use additional articles.

\DeclareAcroArticle{\(\langle\text{name}\rangle\)}{\(\langle\text{default}\rangle\)}

This command can be used to define properties and options analogous to the indefinite article which have been defined this way:

\begin{quote}
\acro{indefinite}{\(a\)}
\end{quote}

In general \DeclareAcroArticle{\(\langle\text{foo}\rangle\)}{\(\langle\text{x}\rangle\)} defines the option

\(\langle\text{foo}\rangle = \{\langle\text{value}\rangle\}\) \hspace{1cm} \text{Initial:} \langle\text{x}\rangle

and these properties

\begin{align*}
\text{short-}\langle\text{foo}\rangle & = \{\langle\text{value}\rangle\} \hspace{1cm} \text{Initial:} \langle\text{x}\rangle \\
\text{alt-}\langle\text{foo}\rangle & = \{\langle\text{value}\rangle\} \hspace{1cm} \text{Initial:} \langle\text{x}\rangle \\
\text{long-}\langle\text{foo}\rangle & = \{\langle\text{value}\rangle\} \hspace{1cm} \text{Initial:} \langle\text{x}\rangle \\
\text{foreign-}\langle\text{foo}\rangle & = \{\langle\text{value}\rangle\} \hspace{1cm} \text{Initial:} \langle\text{x}\rangle \\
\text{single-}\langle\text{foo}\rangle & = \{\langle\text{value}\rangle\} \hspace{1cm} \text{Initial:} \langle\text{x}\rangle \\
\text{extra-}\langle\text{foo}\rangle & = \{\langle\text{value}\rangle\} \hspace{1cm} \text{Initial:} \langle\text{x}\rangle
\end{align*}

In addition another command is defined which is meant to be used in template definitions.

\acro{\text{foo}}

This command tells the template that the article \(\langle\text{foo}\rangle\) should be used.

Section 28 on page 46 has examples of how this can be used to define definite articles and commands that make use of them like this:

\begin{quote}
\dacs{hadopi} \par
\dacl{hadopi}
\end{quote}
26. Translations

For adding additional keywords, or for adding translations to existing keywords, or for changing existing translations ACRO uses this command:

\DeclareAcroTranslation{⟨key⟩}{⟨language=translation list⟩}

With this command new translations keywords can be added and translations for existing keywords can be changed.

\AddAcroTranslations{⟨key⟩}{⟨language=translation list⟩}

Basically the same but this time per language rather than per keyword.

As an example this is how ACRO declares translations for the pages keyword:

```
\DeclareAcroTranslation{pages}{
  Fallback = pp\abbrdot ,
  English = pp\abbrdot ,
  French = pp\abbrdot ,
  German = S\abbrdot ,
  Portuguese = pp\abbrdot
}
```

Translations for a language could be added this way⁹:

```
\AddAcroTranslations{Italian}{
  list-name = Acronimi ,
  page = p\abbrdot ,
  pages = pp\abbrdot ,
  sequens = s\abbrdot ,
  sequentes = ss\abbrdot ,
  also = anche ,
  and = e ,
  or = o
}
```

The existing keywords had been shown in table 1 on page 34.

---

⁹ ACRO already has the translations for Italian.
27. Properties

As you know from section 5 \acro comes with quite a number of predefined properties which control various aspects of acronyms. However, there are cases when additional properties would be nice to have and to use. It can be done with the following command:

\DeclareAcroProperty*?!|\{\langle name\rangle\}
This defines the new property \langle name\rangle. The command has five optional arguments most of which you probably never need.

The optional star * ensures that each acronym gets a unique value for the property.

The optional question mark ? creates a boolean property. That is a property that only can get the values true or false and when it is used without value (not an empty value!) then true is assumed.

The optional exclamation mark ! creates a mandatory property. An error is raised if an acronym does not set it.

The optional pipe | creates a static property which means its value is written to an auxiliary file and read in again at begin document. Once set the value is the same throughout the document.

The optional greater as symbol > creates a display property. This additionally defines the two boolean options \langle name\rangle/display and list/\langle name\rangle/display, both initially set to true. If these options are set to false the acronym commands or the list act as if the property \langle name\rangle has not been set. The foreign property is an example.

\DeclareAcroPropertyAlias*?!|\{\langle name/one.taboldstyle\rangle\}{\langle name/two.taboldstyle\rangle}
This newly declares property \langle name/one.taboldstyle\rangle and makes it an alias of property \langle name/two.taboldstyle\rangle. This means that \langle name/one.taboldstyle\rangle gets the same value that \langle name/two.taboldstyle\rangle has unless it is set explicitly. Property \langle name/two.taboldstyle\rangle must exist.

\MakeAcroPropertyAlias{\langle name/one.taboldstyle\rangle}{\langle name/two.taboldstyle\rangle}
This makes property \langle name/one.taboldstyle\rangle and makes it an alias of property \langle name/two.taboldstyle\rangle. Both properties must exist.

Examples for defining and using new properties are shown in section A, for example, examples 8 or 9.

28. Own acronym commands

28.1. Background

You can define own acronym commands or redefine the existing ones with commands similar to \NewDocumentCommand from the xparse package [L3P].

\NewAcroCommand{\langle command\rangle}{\langle arg. spec.\rangle}{\langle code\rangle}
This creates the new command \langle command\rangle with the argument specification\footnote{in the sense of an xparse command.} so \langle arg. spec.\rangle and
28. Own acronym commands

replacement text (code). There are significant differences to \texttt{NewDocumentCommand}: the new command always has two additional arguments: an optional star and an optional argument for options. You can ignore this fact in your definition, though. However, the command must at least have one argument and the first argument must refer to the ID. Everything else is up to you.

The new command has the suiting framework to recognize trailing tokens, count usage, index, and add a citation if necessary.

\texttt{\RenewAcroCommand\{command\}\{arg.spec\}\{code\}}
Like \texttt{\NewAcroCommand} but redefines an existing command.

\texttt{\UseAcroTemplate\{type\}\{name\}\{argument number\}\{arguments\}}
The argument \texttt{(type)} defaults to \texttt{acronym} and \texttt{(argument number)} defaults to 1. The command must be followed by as many mandatory arguments as you specify with \texttt{(argument number)}. All predefined acronym templates use the first argument as \texttt{id} so they must use one argument.

Let’s see an example. This is the definition of \texttt{\ac}:

```
\NewAcroCommand\ac{m}\{\UseAcroTemplate\{first\}\{#1\}}
```

Equivalent definitions would be:

```
\NewAcroCommand\ac{m}\{\UseAcroTemplate\{acroynm\}\{first\}\{#1\}}
\NewAcroCommand\ac{m}\{\UseAcroTemplate\{acroynm\}\{first\}[1]\{#1\}}
\NewAcroCommand\ac{m}\{\UseAcroTemplate\{first\}\{1\}\{#1\}}
\NewAcroCommand\ac{m}\{\UseAcroTemplate\{first\}\{2\}\{#1\}\{\}}
```

There are a number of switch commands which determine a certain behavior. They tell the following template how to interpret certain conditionals and how to use \texttt{\acrowrite}.

\texttt{\acrocite}
Tells \texttt{ACRO} to output the citation.

\texttt{\acrodonotuse}
Tells \texttt{ACRO} to not count this as usage.

\texttt{\acroplural}
Use plural form.

\texttt{\acroindefinite}
Use indefinite article

\texttt{\acroupper}
Use uppercase form.
\acrofull
Use first or full form.

Here is an example that makes use of them:

```latex
\NewAcroCommand\acs{m}{%
  \acroup\acroindefinite\UseAcroTemplate{short}{#1}%
}\
```

### 28.2. Create commands for possessive endings

Let’s say you want to add an ending for the genitive case. First you define the appropriate ending:

```latex
\DeclareAcroEnding{possessive}{'s'}{'s'}
```

Then you define commands which make use of this ending:

```latex
\NewAcroCommand\acg{m}{\acropossessive\UseAcroTemplate{first}{#1}}
\NewAcroCommand\acsg{m}{\acropossessive\UseAcroTemplate{short}{#1}}
\NewAcroCommand\aclg{m}{\acropossessive\UseAcroTemplate{long}{#1}}
\NewAcroCommand\acfg{m}{%
  \acrofull
  \acropossessive
  \UseAcroTemplate{first}{#1}%
}\
\NewAcroCommand\iacsg{m}{%
  \acroindefinite
  \acropossessive
  \UseAcroTemplate{short}{#1}%
}\
```

You maybe also define acronyms with corresponding properties\footnote{11. Bear with me if this is incorrect: English is not my native language.}:

```latex
\DeclareAcronym{MP}{
  short = MP ,
  long = Member of Parliament ,
  plural-form = Members of Parliament ,
  long-possessive-form = Member’s of Parliament
}\
```
Now you can use it like this:

This is the \ac{MP} first day at work after \dots

This is the MP’s first day at work after …

29. Own **ACRO** style files

When you want to use your definitions regarding **ACRO** repeatedly then it makes sense to put them in a file which you put somewhere in your local LaTeX tree. There are three options:

1. Put them in a simple .tex file in \input it.
2. Put in in a .sty file and include it with \usepackage after **ACRO**.
3. Create a style file following this pattern described below.

```
acro.style.(name).code.tex
```

This file should start with

```
\AcroStyle{name}
```

and input the file with \acsetup using the option

```
load-style = {\langle name\rangle}
```

This is more or less the same as if you’d use the package variant but naturally ensures that you load it after **ACRO** and in the future might provide other bells and whistles, too.

The command

```
\AcroStyle*{\langle style\rangle}\{\langle details\rangle\}
```

has an optional star which switches to expl3 syntax. It also has an optional argument \langle details\rangle with the same purpose and usage as the one from \ProvidesPackage. A typical usage would look like

```
\AcroStyle{abbrev}\{2020/04/21 abbreviations with acro (CN)\}
```

```
\NewDocumentCommand\newabbreviation{mmm}{\DeclareAcronym{#1}{ short = #2 , #3 , class = abbrev , no-index }}
```

```
\NewDocumentCommand\printabbreviations{O{}}{\printacronyms[#1,include=abbrev]}
```

```
\printacronyms[\#1,include=abbrev]
```
Part IV.
Appendix

A. Examples

Example 1: Basic usage
Links: [TeX] [PDF]  File: acro.example.basic.tex

\begin{verbatim}
7 \acsetup{ 
9     make-links ,
10     pages / display = first ,
11     pages / fill = {, }
12 } 
13 \DeclareAcronym{CDMA}{
14
15 \section{Intro}
16
17 In the early nineties, GSM was deployed in many European countries. GSM offered for the first time international roaming for mobile subscribers. The GSM's use of Time Division Multiple Access (TDMA) as its communication standard was debated at length. And every now and then there are big discussions whether Code Division Multiple Access (CDMA) should have been chosen over TDMA.

18 Furthermore
19
20 The reader could have forgotten all the nice acronyms, so we repeat the meaning again.
21
22 If you want to know more about Global System for Mobile communication (GSM), Time Division Multiple Access (TDMA), Code Division Multiple Access (CDMA) and other acronyms, just read a book about mobile communication.
23
24 Just to mention it: There is another used acronym (UA), just for testing purposes!

25 \end{verbatim}

Example 2: Re-implement \acrolike
Links: [TeX] [PDF]  File: acro.example.acflike.tex

\begin{verbatim}
7 \RenewAcroCommand\acrolike{mm}{% 
8     \acroformat{long}{#2} \UseAcroTemplate{short}{#1}%%
9 } 
10 \begin{document}
11 \ac{cd}
12 \acrolike{cd}{Rohling}
13
14 \end{document}
15
16 \begin{quote}
17 Compact Disc (CD)
18 Rohling (CD)
19 \end{quote}

\end{verbatim}
A. Examples

Example 3: Invisible command for backref

```latex
\DeclareAcronym{ny}{
  short = NY ,
  long = New York ,
}
```

Example 4: Defining a definite article

```latex
\NewAcroCommand\dacsm{\acrodefinite
  UseAcroTemplate{short}{#1}}
\NewAcroCommand\dacss{\acroupper\acrodefinite
  UseAcroTemplate{long}{#1}}
\NewAcroCommand\dacl{\acrodefinite
  UseAcroTemplate{long}{#1}}
\NewAcroCommand\Dacs{m}{\acroupper\acrodefinite
  UseAcroTemplate{short}{#1}}
\NewAcroCommand\Dacl{m}{\acrodefinite
  UseAcroTemplate{long}{#1}}
\DeclareAcronym{hadopi}{
  short = HADOPI ,
  long = Haute Autorité pour la diffusion des œuvres et la protection des droits sur Internet
}
```

Example 5: Write the list of acronyms to an external file

```latex
\if@filesw
  \newwrite\acro@list
  \immediate\openout\acro@list\jobname.ac\relax
  \immediate\write\acro@list{\string\begin{description}}
  \let\item\relax
  \acronymsmapF{
    \immediate\write\acro@list{\string\space}\space}
```
A. Examples

Example 6: Insert word between acronym and citation

Links: [TeX] [PDF] [github]  
File: acro.example.issue-154.tex

```
short = UC ,
long = Universal Composability ,
cite = xxx
}
\begin{document}
\We use the \ac{uc}[ model] for dots
```

We use the Universal Composability (UC) model for dots

Example 7: How to define a possessive ending

Links: [TeX] [PDF]  
File: acro.example.possessive.tex

```
\NewAcroCommand\acsg{m}{\acropossessive
UseAcroTemplate{short}{#1}}
\NewAcroCommand\aclg{m}{\acropossessive
UseAcroTemplate{long}{#1}}
\NewAcroCommand\acfg{m}{%
ACROfull
\acropossessive
UseAcroTemplate{first}{#1}%
}
\NewAcroCommand\iacsg{m}{%
```

Member’s of Parliament (MP’s)
Member’s of Parliament
Member’s of Parliament (MP’s)
a MP’s
```

Example 8: Additional alternative form

Links: [TeX] [PDF] [forum]  
File: acro.example.texsx-505891.tex

```
\RenewAcroTemplate[list]{description}{%
\acroheading
\acropreamble
\begin{description}
\acrosmapF{
\item [%
\acrowrite{short}%
\acroifT{alt}{/}\acrowrite{alt}%
```

Acronyms

four another description
one/two/three common
A. Examples

Example 9: Foreign short form

\begin{verbatim}
\RenewAcroTemplate{long-short}{% 
  \acroiffirstTF{\acrowrite{long}\acspace}{(\one\two\three}\acroifT{foreign}{, }}{\one\two\three}\acrowrite{foreign}{%}
\end{verbatim}

Datenschutz-Grundverordnung (DSGVO, “General Data Protection Regulation”, GDPR)

Abkürzungen

DSGVO Datenschutz-Grundverordnung (“General Data Protection Regulation”, GDPR)

Example 10: Species

\begin{verbatim}
\#1 ,
tag = species ,
  first-style= long ,
  format = \itshape 
}%
\newspecies{ecoli}{E. coli}{Escherichia coli}
\end{verbatim}

First use: Escherichia coli
Second use: E. coli

Example 11: Capitalization

\begin{verbatim}
short = 3-APA ,
  long = \iupac{3-azido-1-propyl|amine}
} 
\DeclareAcronym{CuAAC}{
  short = \enquote{click} chemistry ,
  long = copper(I)-catalyzed azide-alkyne cycloaddition 
}
\end{verbatim}

1: 3-Azido-1-propylamine
2: “Click” chemistry
3: Copper(I)-catalyzed

1
### B. Acronyms

Below all abbreviations are listed which have been defined for the manual.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
<th>Page References</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>compact disc</td>
<td>4, 13</td>
</tr>
<tr>
<td>CTAN</td>
<td>Comprehensive \TeX Archive Network</td>
<td></td>
</tr>
<tr>
<td>e.g.</td>
<td>for example (Latin: exempli gratia)</td>
<td></td>
</tr>
<tr>
<td>ECU</td>
<td>Steuergerät (Englisch: Electronic Control Unit)</td>
<td>23</td>
</tr>
<tr>
<td>etc.</td>
<td>et cetera</td>
<td>13, 28</td>
</tr>
<tr>
<td>HADOPI</td>
<td>Haute Autorité pour la diffusion des œuvres et la protection des droits sur l’Internet</td>
<td>44</td>
</tr>
<tr>
<td>ID</td>
<td>identification string</td>
<td>6f., 39, 46f.</td>
</tr>
<tr>
<td>JPEG/JPG</td>
<td>Joint Photographic Experts Group</td>
<td>12</td>
</tr>
<tr>
<td>LA</td>
<td>Los Angeles</td>
<td></td>
</tr>
<tr>
<td>LPPL</td>
<td>\TeX Project Public License</td>
<td></td>
</tr>
<tr>
<td>MP</td>
<td>Member of Parliament</td>
<td>21, 43, 48</td>
</tr>
<tr>
<td>NATO</td>
<td>Organisation des Nordatlantikvertrags (Englisch: North Atlantic Treaty Organization)</td>
<td></td>
</tr>
<tr>
<td>NY</td>
<td>New York</td>
<td>25</td>
</tr>
<tr>
<td>PDF</td>
<td>Portable Document Format</td>
<td>9, 19, 31f.</td>
</tr>
<tr>
<td>PNG</td>
<td>Portable Network Graphics</td>
<td></td>
</tr>
<tr>
<td>PU</td>
<td>Polyurethan</td>
<td></td>
</tr>
</tbody>
</table>
C. References


D. Index
<table>
<thead>
<tr>
<th>Index</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW</td>
<td>21</td>
</tr>
<tr>
<td>T</td>
<td></td>
</tr>
<tr>
<td>tabu</td>
<td>36</td>
</tr>
<tr>
<td>tabu (package)</td>
<td>36</td>
</tr>
<tr>
<td>tabular (template)</td>
<td>36, 40 f.</td>
</tr>
<tr>
<td>tag (property)</td>
<td>7, 16</td>
</tr>
<tr>
<td>Talbot, Nicola L.C.</td>
<td>25</td>
</tr>
<tr>
<td>template</td>
<td>15</td>
</tr>
<tr>
<td>The \TeX{}3 Project Team</td>
<td>9, 31, 46</td>
</tr>
<tr>
<td>threshold</td>
<td>16</td>
</tr>
<tr>
<td>\textbf{title}</td>
<td>25</td>
</tr>
<tr>
<td>toc (template)</td>
<td>36, 39, 41</td>
</tr>
<tr>
<td>translations</td>
<td>33</td>
</tr>
<tr>
<td>translations (package)</td>
<td>33 f.</td>
</tr>
<tr>
<td>\textbf{U}</td>
<td></td>
</tr>
<tr>
<td>UFO</td>
<td>11, 22, 24</td>
</tr>
<tr>
<td>upgrade</td>
<td>5</td>
</tr>
<tr>
<td>use</td>
<td>16 f., 27 f., 31 f.</td>
</tr>
<tr>
<td>use-id-as-short</td>
<td>6</td>
</tr>
<tr>
<td>\UseAcroTemplate</td>
<td>47 f., 50 ff., 54</td>
</tr>
<tr>
<td>\textbf{V}</td>
<td></td>
</tr>
<tr>
<td>version</td>
<td>5</td>
</tr>
<tr>
<td>\textbf{W}</td>
<td></td>
</tr>
<tr>
<td>Wemheuer, Moritz</td>
<td>26</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>25</td>
</tr>
<tr>
<td>\textbf{X}</td>
<td></td>
</tr>
<tr>
<td>xparse</td>
<td>46</td>
</tr>
<tr>
<td>xparse (package)</td>
<td>46</td>
</tr>
</tbody>
</table>