ACRO

v3.8  2022/04/01

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  3
  1. Licence  3
  2. Glossary  3
  3. Reading the manual  3
  4. ACRO for the impatient  4

II. Comprehensive description of creation and usage of acronyms  6
  6. Declaring acronyms and other abbreviations  6
     6.1. Basic properties  7
     6.2. Properties related to plural and indefinite forms  8
     6.3. Properties related to formatting  9
     6.4. Properties related to the created PDF file  10
  7. Using acronyms  11
  8. Alternative short forms  13
  9. The first or full appearance  14
 10. Single appearances of an acronym  15
 11. Printing the list  16
     11.1. The main command and its options  16
     11.2. Add page numbers to the list  17
     11.3. Filter lists using tags  18
     11.4. Local lists  19
 12. Formatting  20
  13. Plural forms and other endings  21
     13.1. The plural ending and the plural form  21
     13.2. Other endings  23
  14. Articles  23
     14.1. Indefinite forms  23
Part I.
Get started with \texttt{ACRO}

1. Licence

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

2. Glossary

This manual repeatedly uses a number of terms which are explained in this section.

\begin{description}
\item[articles] Articles are prefixes to acronyms, usually separated with a blank. Different types of articles are mutually exclusive.
\item[endings] Endings are postfixes to acronyms, usually not separated from the acronym. Different types of endings are mutually exclusive.
\item[load-time option] A load-time \texttt{option} is a package option of \texttt{ACRO} which \texttt{must} be set as option to \texttt{\usepackage\{}\texttt{(options)}\texttt{\}}\texttt{\}{\texttt{acro}}.
\item[module] A \texttt{module} is something like a superordinate category for options. There are several such modules like \texttt{list} or \texttt{pages} for example. Options belonging to a module typically influence certain objects. For instance options belonging to the \texttt{pages} influence how page numbers in the list of acronyms are handled.
\item[option] An \texttt{option} is a package option of \texttt{ACRO} which must set with \texttt{\acsetup}. It \texttt{cannot} be set as option to \texttt{\usepackage}. Options usually also can be set in the optional argument of \texttt{\ac} and friends.
\item[property] A \texttt{property} is an option to the second argument of the \texttt{\DeclareAcroym} command. They are options of an individual acronym if you will.
\item[template] A \texttt{template} determines how different objects of \texttt{ACRO} are printed. This includes the acronyms themselves but also for example the list of acronyms as a whole.
\item[translations] Localisation strings which can be modified.
\end{description}

3. Reading the manual

This manual uses a certain color code in order to be able to distinguish options, properties and templates visually from each other.
option = {⟨value⟩}
Options are displayed in yellow. For the description of options a certain nomenclature is used
which is explained in section 5. Some options belong to module which are displayed in red.

property = {⟨value⟩}
In order to distinguish them visually from options properties are displayed in blue. The nomenclature
of properties is quite similar to the one of the options so it is not explained explicilty but should be clear by analogy.

template
In order to distinguish them visually from options and properties templates are displayed in green.

4. A C R O for the impatient

Acronyms are defined in the preamble via the command
\DeclareAcronym{⟨id⟩}{⟨properties⟩}
where ⟨id⟩ is a unique string to identify the acronym and ⟨properties⟩ is a key/value list of
acronym properties. These include:

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

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

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.
5. Setting options

\texttt{\textbackslash acf}\{\langle id \rangle\} \quad \texttt{\textbackslash Acf}\{\langle id \rangle\}

\texttt{\textbackslash acf} prints the full form of the acronym \langle id \rangle. \texttt{\textbackslash Acf} does the same but uppercases the first letter.

Let’s say you defined CD as follows:

\begin{verbatim}
\DeclareAcronym{cd}{
  short = CD ,
  long = compact disc
}
\end{verbatim}

Then the usage is

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

first compact disc (CD)
second CD
long compact disc
short CD
full compact disc (CD)

5. Setting options

5.1. Load-time options

\texttt{acro} knows only a small set of load-time options which can be used as argument to \texttt{usepackage}:

\begin{itemize}
  \item \texttt{upgrade} = \texttt{true}|\texttt{false} \quad \texttt{Initial: false}
\end{itemize}

When this option is used \texttt{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.

You can still use version 2 by specifying using one of the following options:

\begin{verbatim}
\usepackage{acro}[=v2]
\usepackage{acro}[=version2]
\end{verbatim}

5.2. Setup command

All options of \texttt{acro} that have not been mentioned in section 5.1 have to be set up either with this command

\texttt{\textbackslash 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.

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

\[
\text{option} = \text{choiceA}|\text{choiceB}|\text{choiceC}\quad \text{Initial: choiceB}
\]
An option with a determined set of choices. The underlined value is chosen if the option is
given without value.

\[
\text{option} = \text{true}|\text{false}
\]
A boolean option with the only 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):

\[
\text{\acssetup}{
\begin{array}{l}
\text{make-links = true ,} \\
\text{index ,} \\
\text{format = \textbf{emph} ,} \\
\text{list / local ,} \\
\text{list / display = all}
\end{array}
}\]

Part II.
Comprehensive description of creation and usage of acronyms

6. 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.

\texttt{\textbackslash DeclareAcronym\{\langle id \rangle\}\{\langle list of properties \rangle\}}

The basic command for declaring an acronym where \langle id \rangle 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.
6. Declaring acronyms and other abbreviations

**case-sensitive** = **true|false**  
Initial: true

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.

**case-insensitive** = **true|false**  
Initial: false

The inverse of the option **case-sensitive**.

since v3.6 (2022/01/04)

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

### 6.1. 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.

**extra** = `{⟨text⟩}`  
(initially empty)

Extra information to be added in the list of acronyms.

**foreign** = `{⟨long form in foreign language⟩}`  
(initially empty)

Can be useful when dealing with acronyms in foreign languages, see section 15 on page 24 for details.

**long-post** = `{⟨text⟩}`  
(initially empty)

⟨text⟩ is appended to the long form of the acronym in the text but not in the list of acronyms.

**post** = `{⟨text⟩}`  
(initially empty)

⟨text⟩ is appended to the acronym in the text but not in the list of acronyms.

**single** = `{⟨text⟩}`  
if unused then equal to long

If provided ⟨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 10 on page 15.
6. Declaring acronyms and other abbreviations

\[ \text{sort} = \{ (\text{text}) \} \]

if unused then equal to short
If used the acronym will be sorted according to this property instead of its short form.

\[ \text{tag} = \{ (\text{csv list}) \} \]

(tag(s) of an acronym. (initially empty)
\[ \text{cite} = [ (\text{prenote}) ] [ (\text{postnote}) ] \{ (\text{citation keys}) \} \]

A citation that is printed to the acronym according to an option explained later. (initially empty)
\[ \text{before-citation} = \{ (\text{text}) \} \]

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

This property allows to overwrite the automatic index entry with an arbitrary one. See section 17.2 on page 28 for details. (initially empty)
\[ \text{index-sort} = \{ (\text{text}) \} \]

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 17.2 on page 28 for details.
\[ \text{index-cmd} = \{ (\text{index command}) \} \]

This sets the indexing command for the acronym. If unused then the command set by the corresponding option is used. See section 17.2 on page 28 for details.

6.2. Properties related to plural and indefinite forms

\[ \text{short-plural} = \{ (\text{text}) \} \]

Initial: s
The plural ending appended to the short form. (initially empty)
\[ \text{short-plural-form} = \{ (\text{text}) \} \]

The plural short form of the acronym; replaces the short form when used instead of appending the plural ending. (initially empty)
\[ \text{long-plural} = \{ (\text{text}) \} \]

Initial: s
The plural ending appended to the long form. (initially empty)
\[ \text{long-plural-form} = \{ (\text{text}) \} \]

Plural long form of the acronym; replaces the long form when used instead of appending the plural ending. (initially empty)
\[ \text{alt-plural} = \{ (\text{text}) \} \]

Initial: s
The plural ending appended to the alternative form. (initially empty)
\[ \text{alt-plural-form} = \{ (\text{text}) \} \]

The plural alternative form of the acronym; replaces the alternative form when used instead of appending the plural ending. (initially empty)
6. Declaring acronyms and other abbreviations

foreign-plural = \{text\}

The plural ending appended to the foreign form.

initial: s

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.

initial: a

long-indefinite = \{text\}

Indefinite article for the long form.

initial: a

alt-indefinite = \{text\}

Indefinite article for the alternative form.

initial: a

6.3. Properties related to formatting

format = \{code\}

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

(initially empty)

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.

eextra-format = \{code\}

The format used for the additional information of the acronym.

(initially empty)

foreign-format = \{code\}

The format used for the foreign form of the acronym.

(initially empty)

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

The style of the first appearance of the acronym, see also section 9 on page 14.

(initially empty)

subsequent-style = long-short|short-long|short|long|footnote

The style of the appearance of the acronym after the first time.

since v3.4

(2020/12/25)
6. Declaring acronyms and other abbreviations

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

The style of a single appearance of the acronym, see also section 10 on page 15.

### 6.4. Properties related to the created PDF file

\texttt{pdfstring} = \langle pdfstring \rangle \hspace{1cm} \text{if unused then equal to short}

Used as PDF string replacement in bookmarks when used together with the hyperref [ORT21] or the bookmark package [Obe20].

\texttt{pdfcomment} = \langle \text{\texttt{text}} \rangle

Sets a tooltip description for an acronym. For actually getting tooltips you also need an appropriate setting of the options \texttt{pdfcomment/cmd} and \texttt{pdfcomment/use}, see also section 21.3 on page 33.

\texttt{short-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to short}

Sets the ActualText property as presented by the accsupp package for the short form of the acronym.

\texttt{long-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to long}

Sets the ActualText property as presented by the accsupp package for the long form of the acronym.

\texttt{alt-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \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.

\texttt{foreign-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to foreign}

Sets the ActualText property as presented by the accsupp package for the foreign form of the acronym.

\texttt{extra-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to extra}

Sets the ActualText property as presented by the accsupp package for the extra information of the acronym.

\texttt{single-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to long-acc}

Sets the ActualText property as presented by the accsupp package for a single appearance of the acronym.

\texttt{list-acc} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to list}

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

### 6.5. Further properties

\texttt{list} = \langle \text{\texttt{text}} \rangle \hspace{1cm} \text{if unused then equal to long}

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

\textbf{foreign-babel} = \{(\textit{language})\}
\hspace{1cm} \text{(initially empty)}

The babel [Bra22] or polyglossia [Cha21] language of the foreign form. This language is used to wrap the entry with \texttt{\foreignlanguage{\langle 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.

\textbf{foreign-locale} = \{(\textit{language})\}
\hspace{1cm} \text{(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 15 on page 24 for details.

\textbf{preset} = \{(\textit{set name})\}
\hspace{1cm} \text{(initially empty)}

\textit{since v3.5} (2021/01/16)

Enables to load a set of properties that has been defined earlier with \texttt{\NewAcroPreset}, see section 6.6.

\textbf{uselist} = \{(\textit{csv list of acronym ids})\}
\hspace{1cm} \text{(initially empty)}

\textit{since v3.5}

If this property is given and all acronyms specified in this property have been used before the first time the current acronym is used it behaves as if it has been used before.

6.6. Presets

\textit{since v3.5}

Sometimes it can be useful to have different kinds of acronyms or abbreviations or similar which share a common set of properties. Such sets can be defined with these commands:

\texttt{\NewAcroPreset\{(set name)\}\{\textit{csv list of properties}\}}

Defines the property set \texttt{\langle set name\rangle}. Any valid property can be set in \texttt{\langle csv list of properties\rangle}.

\texttt{\RenewAcroPreset\{(set name)\}\{\textit{csv list of properties}\}}

Redefines the property set \texttt{\langle set name\rangle}.

\texttt{\DeclareAcroPreset\{(set name)\}\{\textit{csv list of properties}\}}

Defines or redefines the property set \texttt{\langle set name\rangle} without checking.

7. 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{\acronym\{\textit{options}\}\{\textit{id}\}}

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

\texttt{\ac\{\textit{id}\}} \texttt{\Ac\{\textit{id}\}} \texttt{\acp\{\textit{id}\}} \texttt{\Acp\{\textit{id}\}} \texttt{\iac\{\textit{id}\}} \texttt{\Iac\{\textit{id}\}}

\texttt{\ac} prints the acronym \texttt{\langle 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
7. Using acronyms

at the beginning of a sentence. The commands \acs and \Acs, resp., print the corresponding plural forms. The commands \iac and \Iac, resp., print indefinite forms.

\acs\{⟨id⟩\} \Acs\{⟨id⟩\} \acsp\{⟨id⟩\} \Iacs\{⟨id⟩\}
\acs prints the short form of the acronym ⟨id⟩. \Acs does the same but uppercases the first letter. The commands \acsp and \Acs, resp., print the corresponding plural forms. The commands \iacs and \Iacs, resp., print indefinite forms.

\acl\{⟨id⟩\} \Acl\{⟨id⟩\} \aclp\{⟨id⟩\} \Iacl\{⟨id⟩\}
\acl prints the long form of the acronym ⟨id⟩. \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\{⟨id⟩\} \Aca\{⟨id⟩\} \acap\{⟨id⟩\} \Acap\{⟨id⟩\} \iaca\{⟨id⟩\} \Iaca\{⟨id⟩\}
\aca prints the alternative short form of the acronym ⟨id⟩. \Aca does the same but uppercases the first letter. The commands \acap and \Acap, resp., print the corresponding plural forms. The commands \iaca and \Iaca, resp., print indefinite forms.

\acf\{⟨id⟩\} \Acf\{⟨id⟩\} \acfp\{⟨id⟩\} \Acfp\{⟨id⟩\} \iacf\{⟨id⟩\} \Iacf\{⟨id⟩\}
\acf prints the full form of the acronym ⟨id⟩. \Acf does the same but uppercases the first letter. The commands \iacfp and \Iacf, 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:

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

The typical outputs look like this:

\ac\{ufo\} \iac\{ufo\} \iacl\{ufo\} \Iacf\{ufo\} \acfp\{ufo\}

unidentified flying object (unbekanntes Flugobjekt, UFO)
a UFO
an unidentified flying object
8. 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 `alt` is there for.

```latex
alt = {⟨text⟩}
```

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:

```
\ac{jpg} \\n\ac{jpg} \\n\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 25 on page 38 and section 9 on the next page. The alternative form is also printed in the list of acronyms, see section A on page 58. This can also be changed by altering the template for the list, again see section 25.
9. The first or full appearance

If an acronym is used for the first time with `\ac` (after any number of usages with the starred forms of the usage commands listed in section 7 on page 11) or if an acronym is used `\acf`, then the first or full appearance of the acronym is printed.¹

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

```
\def\firststyle{long-short|short-long|short|long|footnote}
```

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 acronym. All pre-defined templates can be found in section 25.1 on page 38.

**since v3.4**

The style of the appearance of the acronym after the first time. This options sets the appearance for all acronyms. Available options in reality are the names of all defined templates of the type acronym. All pre-defined templates can be found in section 25.1 on page 38.

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

```
\def\firststyle{long-short|short-long|short|long|footnote}(initially empty)
```

The style of the first appearance of the acronym.

Let’s again look at an example:

```
1. \Acf[first-style=long-short]{cd} \\ compact disc (CD)
2. \Acf[first-style=short-long]{cd} \\ CD (compact disc)
3. \Acf[first-style=footnote]{cd} \\ CDₐ compact disc
4. \Acf[first-style=long]{cd} \\ CD
5. \Acf[first-style=short]{cd} \\ a. compact disc
```

This also demonstrates the use of the optional argument.

An example of an abbreviation that should have `long` as first appearance might be “etc.”, defined like this

```
1. \DeclareAcronym{etc}{
2.  short = etc\acdot ,
3.  long = et cetera ,
4.  format = \textit ,
5.  first-style = long ,
6.  plural =
7. }
```

---

¹. This usually requires at least two \LaTeX\ runs until it is stable.
10. Single appearances of an acronym

and output like this:

```
\% etc, \%etc, \%etc.

et cetera, etc. etc.
```

The command `\acdot` is explained in section 19 on page 30. Basically it checks if a dot follows and outputs a dot if not.

10. 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 7 on page 11), then the single appearance of the acronym is printed. If an acronym is used only once (not counting usages with the starred forms of the usage commands listed in section 7 on page 11), then the single appearance of the acronym is printed. If an acronym is used only once (not counting usages with the starred forms of the usage commands listed in section 7 on page 11), then the single appearance of the acronym is printed.

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

```
single = true|false|(number)  
```

Initial: `false`

This option determines whether a single appearance of an acronym counts as usage. It might be desirable 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 `(number)` the minimal number of usages can be given that needs to be exceeded. `single = {1}` is the same as `single = {true}`.

```
single-style = long-short|short-long|short|long|footnote  
```

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 `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 `acronym`. All pre-defined templates can be found in section 25.1 on page 38.

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

```
single = {⟨text⟩}  
```

(initially empty)

If provided `⟨text⟩` will be used instead of whatever template ("style") has been set for the single appearance if the acronym is only used a single time and the option `single` has been set.

```
single-style = long-short|short-long|short|long|footnote  
```

(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:

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

---

2. This usually requires at least two \TeX runs until it is stable.

3. Actually the template `single` is used which typesets the `single` property.
11. Printing the list

And it is used only once in this manual⁴:

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

- PNG

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

11. Printing the list

11.1. The main command and its options

The main idea is simple: just place

\texttt{\textbackslash printacronyms\{\textit{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 \texttt{ACRO} requiring to re-run.

The options controlling the list are these:

- \texttt{\textbackslash list/template = \textit{description}|\textit{tabular}|\textit{longtable}|\textit{supertabular}|\textit{tabularray}|\textit{lof}|\textit{toclist}}
  - Choose the template to create the list with. See more on this in sections 25 on page 38 and A on page 54.

- \texttt{\textbackslash list/sort = \textit{true}|\textit{false}}
  - Initial: \textit{true}
  - Decide whether to sort the list of acronyms alphabetically or to print it in order of definition.

- \texttt{\textbackslash list/display = \textit{all}|\textit{used}}
  - Initial: \textit{used}
  - Decide whether to print only the acronyms actually used in the document or all acronyms which have been declared in the preamble.

- \texttt{\textbackslash list/exclude = \{}\textit{\textbackslash csv list of tags}\}\texttt{\}}
  - (initially empty)
  - Set a list of tags to exclude from the list. Only acronyms not belonging to one of these tags will be included.

- \texttt{\textbackslash list/include = \{}\textit{\textbackslash csv list of tags}\}\texttt{\}}
  - (initially empty)
  - Set a list of tags to include in the list. Only acronyms belonging to one of these tags will be included.

- \texttt{\textbackslash list/add = \textit{true}|\textit{false}}
  - (initially empty)
  - Since v3.4
  - Set a list of tags to include in the list. These acronyms will be included in any case.

- \texttt{\textbackslash list/heading = \textit{none}|\textit{section}|\textit{section*}|\textit{chapter}|\textit{chapter*}}
  - 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.

---

⁴ You will find it in the list of acronyms in section A nonetheless as this document does \texttt{\textbackslash list/display = \{}\textit{all}\}\texttt{\}}.
11. Printing the list

\texttt{list/name} = \{(\texttt{text})\} \quad \text{Initial: } \acrotranslate{list-name}

Overwrites the text which is used in the heading.

This only has an effect if the list template supports heading templates \textit{and} the heading templates support it. All pre-defined heading templates \textit{do} support this.

\texttt{list/preamble} = \{(\texttt{text})\} \quad \text{(initially empty)}

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 \textit{do} support it.

\texttt{list/locale/display} = \{\texttt{true} | \texttt{false}\} \quad \text{Initial: } \texttt{false}

This options 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 \textit{do} support them.

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

\begin{verbatim}
\acsetup{list/display=all,list/exclude=units}
or
\printacronyms[display=all,exclude=units]
\end{verbatim}

11.2. 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:

\texttt{pages/display} = \{\texttt{first} | \texttt{all} | \texttt{none}\} \quad \text{Initial: } \texttt{none}

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

\texttt{pages/seq/use} = \{\texttt{true} | \texttt{false}\} \quad \text{Initial: } \texttt{true}

Turns a two-page range into \texttt{\langle num\rangle f. (sequens)} and a three-page range into \texttt{\langle num\rangle ff. (sequentes)} when set to \texttt{true}.

\texttt{pages/seq/pre} = \{(\texttt{code})\} \quad \text{Initial: } \texttt{,}

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

\texttt{pages/seq/threshold} = \{(\texttt{num})\} \quad \text{Initial: 3}

The threshold for a page range to be turned into \textit{sequentes}. A page range above the threshold is still typeset as a range: \texttt{\langle num1\rangle – \langle num2\rangle}.

\texttt{pages/fill} = \{(\texttt{code})\} \quad \text{Initial: } \acrodotfill

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

```
\acrodotfill
```

Since v3.5

Creates a dotted line like those in the table of contents. If the macro `\cftdotfill` is defined it is equal to `\cftdotfill{\cftdotsep}`.

```
\acsetup{\printacronyms[pages={display=all,seq/use=false}]}
```

Since v3.3

(2020/11/21)

Additionally to setting these options with `\acsetup` they can be set as options to `\printacronyms`:

### 11.3. 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⟩}  
\\ (initially empty)
```

The tag(s) of an acronym.

```
list/exclude = {⟨csv list of tags⟩}  
\\ (initially empty)
```

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⟩}  
\\ (initially empty)
```

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

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

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

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

\printacronyms[exclude=city]

or print a list with only these acronyms with

\printacronyms[include=city,heading=none]

LA Los Angeles
NY New York .................................................................27

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

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

would only print acronyms with tag c.

11.4. 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

\texttt{\textbackslash barriers/use = true|false} \hspace{1in} Initial: false
this option to true, place \texttt{\textbackslash acbarrier}

before a new chapter starts (this is not necessary for the first chapter), and use \texttt{\textbackslash printacronyms}

with the option

\texttt{\textbackslash list/local = true|false} \hspace{1in} Initial: false
or set this option once in the preamble with \texttt{\textbackslash acsetup} so it is applied to every list.

Please read more on barriers in section 18 on page 29.

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 anstead 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.
12. Formatting

`acro` has a number of options and parameters which can be used to influence the formatting of acronyms.

format = \{(code)\} (initially empty)
Sets the format for both the short and the long form.

format/short = \{(code)\} (initially empty)
Sets the format for the short form.

format/long = \{(code)\} (initially empty)
Sets the format for the long form.

format/first-long = \{(code)\} (initially empty)
Sets the format for the first appearance of the long form.

format/alt = \{(code)\} (initially empty)
Sets the format for the alternative form.

format/extra = \{(code)\} (initially empty)
Sets the format for the extra information.

format/foreign = \{(code)\} (initially empty)
Sets the format for the foreign form.

format/list = \{(code)\} (initially empty)
Sets the format for the long form in the list form.

While this options influence the formatting of the acronyms globally you can also give each acronym its own formatting individually:

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.
13. **Plural forms and other endings**

*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 9 on page 14.

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

The style of a single appearance of the acronym, see also section 10 on page 15.

Per default the individual formatting instructions replace the global ones. This can be changed through the option

*format/replace = true|false*  

Initial: true

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

Let’s see an example:

```
1 \DeclareAcronym{pdf}{
2  short = pdf ,
3  long = Portable Document Format ,
4  short-format = \scshape
5 }
```

```
1 \acsetup{format = \itshape}
2 \acf{pdf} \par
3 \acsetup{format/replace=false}
4 \acf{pdf}
```

*Portable Document Format (PDF)*  
*Portable Document Format (PDF)*

13. **Plural forms and other endings**

13.1. **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.5 This is why ACRO has quite a number of different

5. German is full of such examples.
13. Plural forms and other endings

properties related to plural forms or endings:

**short-plural** = {⟨text⟩}  
Initial: s
The plural ending appended to the short form.

**short-plural-form** = {⟨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⟩}  
Initial: s
The plural ending appended to the long form.

**long-plural-form** = {⟨text⟩}  
(initially empty)
Plural long form of the acronym; replaces the long form when used instead of appending the plural ending.

**alt-plural** = {⟨text⟩}  
Initial: s
The plural ending appended to the alternative form.

**alt-plural-form** = {⟨text⟩}  
(initially empty)
The plural alternative form of the acronym; replaces the alternative form when used instead of appending the plural ending.

**foreign-plural** = {⟨text⟩}  
Initial: s
The plural ending appended to the foreign form.

**foreign-plural-form** = {⟨text⟩}  
(initially empty)
Plural foreign form of the acronym; replaces the foreign form when used instead of appending the plural ending.

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

**short-plural-ending** = {⟨text⟩}  
Initial: s
Defines the plural ending for the short forms to be ⟨text⟩.

**long-plural-ending** = {⟨text⟩}  
Initial: s
Defines the plural ending for the long forms to be ⟨text⟩.

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

```latex
\DeclareAcronym{sw}{
  short = SW ,
  long = Sammelwerk ,
  long-plural = e
}
\DeclareAcronym{MP}{
  short = MP ,
  long = Member of Parliament ,
}```
14. Articles

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{align*}
\texttt{\textbackslash acfp\{sw\} \par} & \text{Sammelwerke (SWs)} \\
\texttt{\textbackslash acfp\{MP\}} & \text{Members of Parliament (MPs)}
\end{align*}

13.2. Other endings

Besides plural endings there are other ones like the genitive case, for example. This is why \texttt{ACRO} generalized the concept. Section 26 on page 46 explains in detail how to define and use additional endings.

14. Articles

14.1. 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.}

\begin{align*}
\texttt{\textbackslash acreset\{ufo\}\%} & \text{a unidentified flying object (unbekanntes Flugobjekt, UFO)} \\
\texttt{a \textbackslash ac\{ufo\} \par} & \text{an UFO} \\
\texttt{an \textbackslash ac\{ufo\}} & \text{a UFO}
\end{align*}

And what good would it be to use a package like \texttt{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 12. We then can just use the dedicated commands and let them decide for us:

\begin{align*}
\texttt{\textbackslash acreset\{ufo\}\%} & \text{an unidentified flying object (unbekanntes Flugobjekt, UFO)} \\
\texttt{\textbackslash iac\{ufo\} \par} & \text{an UFO} \\
\texttt{\textbackslash iac\{ufo\}} & \text{a UFO}
\end{align*}

The commands which also output the indefinite article all start with an “i” and have all been described in section 7 on page 11 already: \texttt{\iac}, \texttt{\Iac}, \texttt{\iacs}, \texttt{\Iacs}, \texttt{\iacl}, \texttt{\Iacl}, \texttt{\iaca}, \texttt{\Iaca}, \texttt{\iacf}, and \texttt{\Iacf}.

\footnote{This may very well be a language specific issue.}
14.2. 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 27 explains in detail how to define and use additional articles.

15. 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:

- `foreign = {{long form in foreign language}}` (initially empty)
  Can be useful when dealing with acronyms in foreign languages, see section 15 for details.

- `foreign-plural = {{text}}` Initial: s
  The plural ending appended to the foreign form.

- `foreign-plural-form = {{text}}` (initially empty)
  Plural foreign form of the acronym; replaces the foreign form when used instead of appending the plural ending.

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

- `foreign-babel = {{language}}` (initially empty)
  The babel or polyglossia language of the foreign form. This language is used to wrap the entry with \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.

- `foreign-locale = {{language}}` (initially empty)
  The language name that is output when the option locale/display is used. If this property is not set then the appropriate value might be derived from foreign-babel.

There are also some options:

- `foreign/display = true|false` Initial: true
  Determine whether to hide or display the foreign form.

- `list/foreign/display = true|false` Initial: true
  Since v3.2 (2020/05/10)
  Determine whether to hide or display the foreign form in the list of acronyms.

- `locale/display = true|false` Initial: false
  This option determines whether the language of the foreign form is printed or not when the full form of the acronym is printed.

- `list/locale/display = true|false` Initial: false
  The same but for the list of acronyms.
15. Foreign language acronyms

\textit{locale/format} = \{(\text{code})\}

Initial: \textit{\texttt{text\_titlecase\_first\_n}}

Determines how said language is formatted when printed. The last command in \{(\text{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 “locale” field by itself:

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

\begin{verbatim}
\acsetup{locale/display,first-style=short-long}
\acf{eg}
\end{verbatim}
16. 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 7. The usage is self-explaining:

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

There are a number of options to control the uppercasing behavior:

- **uppcase/first**: The default setting. Converts the first letter to uppercase.
- **uppcase/title**: This is just a synonym of `first`.
- **uppcase/all**: Converts all letters to uppercase.
- **uppcase/none**: Converts all letters to lowercase
- **uppcase/cam = \{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.
17. Citing and indexing

example you could use `\textsc{capitalisewords}` from the package `mfirstuc` [Tal21]. 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:

\texttt{upercase/short} = \texttt{true|false} 
Initial: true

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

## 17. Citing and indexing

### 17.1. 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:

\begin{verbatim}
\DeclareAcronym{ny} {
  short = NY ,
  long = New York ,
  plural = ,
  tag = city ,
  cite = NewYork
}
\end{verbatim}

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

\begin{verbatim}
\ac{ny} New York (NY) [Wik20]
\end{verbatim}

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} = \{\langle citation command\rangle\} 
Initial: \texttt{\cite}

Choose the command with which citations are printed.

\texttt{cite/group} = \texttt{true|false} 
Initial: 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|all|none} 
Initial: first

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

\texttt{cite/pre} = \{\langle text\rangle\} 
Initial: \texttt{\nobreakspace}

Arbitrary code directly output before the citation.
17. Citing and indexing

**cite/group/cmd** = \{citation command\}  
Initial: `\cite`

Choose the command with which grouped citations are printed.

**cite/group/pre** = \{text\}  
Initial: `,`

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

If for example you use biblatex’s `authoryear` style [Leh+22] 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)

### 17.2. Indexing

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

**index/use** = true|false|indexed  
Initial: false

Enable indexing. If indexed is chosen only the acronyms for which the property `index` has been set are indexed. With true all acronyms are indexed.

**index/cmd** = \{index command\}  
Initial: `\index`

Choose a command for indexing.

**index/disable** = \{code\}  
Initial: `{}`

Sometimes it is desirable to change the meaning of a command inside an index entry. For the entries created by \texttt{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.
18. Barriers

index = \{text\}  
(initially empty)

This property allows to overwrite the automatic index entry with an arbitrary one.

index-sort = \{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.

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 58 that has been used in this manual is also listed in the index.

18. 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 10 on page 15. 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.
19. Trailing tokens

19.1. 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” (\babelhyphen).

You have seen an example for this already:

\begin{verbatim}
\DeclareAcronym{etc}{
  short = etc\acdot ,
  long = et cetera ,
  format = \textit ,
  first-style = long ,
  plural =
}
\end{verbatim}

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

\begin{verbatim}
\ac{etc} and \ac{etc}.
\end{verbatim}

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

\begin{verbatim}
\DeclareAcronym{PU}{
  short = PU ,
  long = Polyurethan ,
  long-plural = e
}
\end{verbatim}

and you use it the first time like this: \ac{PU}-Hartschaum then according to German orthography and typesetting rules this should be printed as

“Polyurethano(\ac{PU})-Hartschaum”

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

\begin{verbatim}
\acf{PU}-Hartschaum Polyurethano(\ac{PU})-Hartschaum
\end{verbatim}

This works because the template long-short\footnote{The template that is used by default for the first appearance.} uses \acs at the appropriate place and the manual setup does
19. Trailing tokens

\acsetup{trailing/activate = dash}

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

19.2. How does it work?

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

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

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

\texttt{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{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}
\acsetup{
  trailing/define = . \{dot\} ,
  trailing/define = {, \{comma\}} ,
  trailing/define = - \{dash\}
  trailing/define = \textbackslash babelhyphen \{babel-hyphen\} ,
  trailing/activate = \{dot,comma\}
}
\end{verbatim}

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

\texttt{\aciftrailing{\langle csv list of token names \rangle}{\langle true \rangle}{\langle false \rangle}}

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

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

\texttt{\acdot}

Inserts \abbrdot if no dot follows.

\texttt{\acspace}

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

\texttt{\abbrdot}

Inserts .@\textbackslash

The definitions are equivalent to the following code:
20. 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 \texttt{tabularx} or \texttt{ltxtable}.

21. Bookmarks, backlinks and accessibility support

21.1. Backlinks

When \texttt{acro} is used together with the package \texttt{hyperref} [ORT21] then you can make use of the following option:

\make-links = \texttt{true}|false Initial: \texttt{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.

\link-only-first = \texttt{true}|false Initial: \texttt{false}

since v3.5

If this is activated in addition to \make-links then \textit{only the first} short or alternative appearance of an acronym will be linked to its description in the list of acronyms.
21. Bookmarks, backlinks and accessibility support

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

21.2. Bookmarks

Since bookmarks (which are created by the hyperref or the bookmark packages [Obe20]) 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\}\quad \text{if unused then equal to short}

Used as PDF string replacement for the short form in bookmarks when used together with the hyperref [ORT21] or the bookmark package [Obe20].

This is for acronyms like

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

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

21.3. PDF comments

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

\texttt{pdfcomments/use = true|false}\quad \text{Initial: false}

This enables the creation of PDF comments.

\texttt{pdfcomments/cmd = \{\textit{code}\}}\quad \text{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:

\texttt{pdfcomment = \{\textit{text}\}}

Sets a tooltip description for an acronym.
### 21. Bookmarks, backlinks and accessibility support

#### 21.4. Accessibility support

**acro** supports the accsupp package [Obe19] when you *also load* hyperref. Then **acro** uses

```latex
\BeginAccSupp{ method = pdfstringdef , ActualText = {PDF} }
\textsc{pdf}\%
\EndAccSupp{}
```

for an acronym defined like this:

```latex
\DeclareAcronym{pdf}{
  short = pdf ,
  long = Portable Document Format ,
  short-format = \textsc{pdf} ,
  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:

- **accsupp/use** = *true*|*false*  
  Initial: *true*  
  When this is true and the package accsupp is loaded then accessibility support is used.

- **accsupp/options** = {{text}}  
  (initially empty)  
  Additional option to be passed to \BeginAccSupp. See the accsupp manual for possible settings.

- **accsupp/method** = {{method}}  
  Initial: pdfstringdef  
  The method used by \BeginAccSupp. 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:

- **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.
22. Localisation

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

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

list-acc = \{⟨text⟩\}
                    \text{if unused then equal to list}
Sets the ActualText property as presented by the accsupp package for the appearance in the list of acronyms.
\end{verbatim}

Extra care has to be taken for plural forms as these can not be picked up automatically right now. You have to explicitly set them for the accessibility support, too:

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

Please note that this mechanism is disabled for inner acronyms if you have nested definitions.

22. 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 [Nie22].

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|⟨language⟩} \hspace{1cm} \text{Initial: auto}

The default setting 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.
Table 1: Available translation keywords.

<table>
<thead>
<tr>
<th>Key</th>
<th>English</th>
<th>French</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>list-name</td>
<td>Acronyms</td>
<td>Acronymes</td>
<td>Abkürzungen</td>
</tr>
<tr>
<td>page</td>
<td>p.</td>
<td>p.</td>
<td>S.</td>
</tr>
<tr>
<td>pages</td>
<td>pp.</td>
<td>p.</td>
<td>S.</td>
</tr>
<tr>
<td>sequens</td>
<td>f.</td>
<td>sq.</td>
<td>f.</td>
</tr>
<tr>
<td>sequentes</td>
<td>ff.</td>
<td>sqq.</td>
<td>ff.</td>
</tr>
<tr>
<td>also</td>
<td>also</td>
<td>aussi</td>
<td>auch</td>
</tr>
<tr>
<td>or</td>
<td>or</td>
<td>ou</td>
<td>oder</td>
</tr>
<tr>
<td>and</td>
<td>and</td>
<td>et</td>
<td>und</td>
</tr>
</tbody>
</table>

\acro only provides support for a handful of languages. You can easily teach \acro your language – see section 28 on page 48 – if it isn’t supported, yet.

\acrotranslate{⟨key⟩}
This command fetches the translation of ⟨key⟩ 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.

23. Patches

In several situations it can lead to wrong results if \acro marks an acronym as used too early or at all. This is why it is possible to disable the mechanism which is responsible:

\acswitchoff
This disables the mechanism which marks acronyms as used. After this command every acronym command like \ac acts like its starred version.

\acswitchon
This command enables the mechanism again.

In certain circumstances \acro uses these commands itself. For example it is often preferable that acronyms are not counted as used in floats, the table of contents or the lists of figures and tables. This is why \acro turns the mechanism off in these places.

Certain table environments typeset their contents twice for measurement purposes. \acro tries to disable the usage mechanism during these phases. The same is true for single line captions from the caption package.

All these patches can be turned off:

\patch/\floats = true|false

Initial: true

En-/disable the \floats patch.

8. 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.
Part III.
Extending **ACRO**

24. Background

24.1. 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 \ac and friends to the list of acronyms. **ACRO** uses three types of templates:

**acronym** These templates can be used to define acronym commands, see section 30 on page 50.

**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 25 on the following page.
24.2. 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 14.2 on page 24.

**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 13.2 on page 23.

**properties** You have already learned about properties. It is possible to define and add further acronym properties to \acro. This is explained in section 29 on page 49.

**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 22 on page 35.

25. Templates

25.1. Pre-defined templates

25.1.1. 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. This template also contains a command \acroendfootnote since v3.7 at the end of the footnote text which in its default definition does nothing.

**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.

**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.
25. Templates

**single**
A template which is used when the property `single` has been set and the option `single` has been set and if the acronym is only used a single time.

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

25.1.2. List templates

**description**
The default list style which places the short form in the item of a `description` environment and adds 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 [Car21] loaded.

**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.

The templates `tabular`, `longtable`, and `supertabular` support the following option:

```
templates/colspec = {{value}}
```

Initial: `>{\bfseries}{.7\linewidth}`

**New**
It enables you to easily adjust the column specification of the standard table templates to your needs. *Please be aware that you need exactly two columns.*

25.1.3. Heading templates

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

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

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

25.2. Defining new templates

For the definition of templates these commands are available:

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

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

\SetupAcroTemplate\{\langle type\rangle\}\{\langle name\rangle\}\{\langle code\rangle\}
since v3.2
Adds \langle code\rangle to the beginning of the template \langle name\rangle of type \langle type\rangle. The default type is acronym.

\SetupNextAcroTemplate\{\langle type\rangle\}\{\langle name\rangle\}\{\langle code\rangle\}
since v3.2
Adds \langle code\rangle to the beginning of the next use of the template \langle name\rangle of type \langle type\rangle. The default type is acronym.

* $\AcroTemplateType$
Within a template this expands to the \langle type\rangle of the current template.

* $\AcroTemplateName$
Within a template this expands to the \langle name\rangle 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.
25. 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.

25.3.1. 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 30 on page 50.

\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⟩}

since v3.1
(2020/05/03)

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.
25. Templates

\acroiftag\{tag\}\{true\}\{false\}
Checks if the current acronym has been given the tag \textit{tag} and either leaves \textit{true} or \textit{false} in the input stream.

\acroifstarred\{true\}\{false\}
since v3.5
Checks if the current call of the acronym is a starred command or not and either leaves \textit{true} or \textit{false} in the input stream.

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

\acroAcronymsMap\{code\}
since v3.5
Maps over all defined acronyms. Within \textit{code} you can refer to the current property with \#1 or with \AcronymID.

\acroMapBreak
since v3.5
Stops the map \acroAcronymsMap and is usually used in combination with a boolean check.

\acroPropertiesSet\{id\}\{csv list of properties\}
since v3.5
Allows the setting of properties of acronym \textit{id} outside of \texttt{declareAcronym}.

25.3.2. Commands for usage in acronym templates

\*\acroifused\{true\}\{false\}
Checks if the current acronym has been used before and either leaves \textit{true} or \textit{false} in the input stream.

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

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

\acrogroupcite

25.3.3. Commands for usage in list templates

\*\acroifchapter\{true\}\{false\}
This just check if \texttt{chapter} is defined. Used in the \texttt{toc} template.

\*\acroifpages\{true\}\{false\}
This is \textit{true} if the option \texttt{pages/display} is set, \textit{and} the current acronym is not single, \textit{and} has at least one page number. \textit{false} otherwise.
25. Templates

\acropages{\langle \text{first} \rangle}{\langle \text{range} \rangle}
If $\text{acroifpages} \text{TF}$ would be $\langle \text{false} \rangle$ this would do nothing. Otherwise, if \text{pages/display} is first it prints the first page number, preceded by $\langle \text{first} \rangle$ if \text{pages/name} is true. If \text{pages/display} is all it prints the page range, preceded by $\langle \text{range} \rangle$ if \text{pages/name} is true.

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

\acroneedpages
since v3.4
This enables the page number displayed. Used in the \textit{toc} and \textit{lof} templates.

\acropagefill
If $\text{acroifpages} \text{TF}$ would be $\langle \text{false} \rangle$ this would do nothing. Otherwise it prints whatever is set by \text{pages/fill}.

\acronymsmap{\langle \text{code} \rangle}
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 $\langle \text{code} \rangle$ #1 refers to the current ID of the acronym. Also $\AcronymID$ expands to the current ID. The latter is important for all the commands that check or print properties of acronyms.

\acronymsmapTF{\langle \text{code} \rangle}{\langle \text{true} \rangle}{\langle \text{false} \rangle}
This does the same as \acronymsmap and also leaves $\langle \text{true} \rangle$ in the input stream if the list is not empty and $\langle \text{false} \rangle$ otherwise. This is useful to trigger a rerun warning.

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

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

\AcroNeedPackage{\langle \text{package} \rangle}
Checks if the package $\langle \text{package} \rangle$ is loaded and throws an error otherwise.

\AcroRerun
Triggers \acro to throw an “empty list” rerun warning.

25.4. New acronym templates
Some templates are quite short and self-explaining:

\begin{verbatim}
\NewAcroTemplate{\langle \text{short} \rangle}{\acrowrite{\langle \text{short} \rangle}}
\end{verbatim}

Some are a little bit more elaborate:
25. Templates

And some templates need to do a lot more:

25.5. New list templates

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

\begin{verbatim}
\NewAcroTemplate[list]{tabular}{% 
\AcroNeedPackage{array}% 
\acronymssmapF{% 
 \AcroAddRow{ 
 \acrowrite{short}% 
 & \acroifT{alt}{/\acrowrite{alt}}% 
 & \acrowrite{list}% 
 \acroifanyT{foreign,extra}{ (}% 
 \acroifT{foreign}{\acrowrite{foreign}\acroifT{extra}{, }}% 
 \acroifT{extra}{\acrowrite{extra}}% 
 \acroifanyT{foreign,extra}{)}% 
 \acropagefill% 
 \acropages{\acrotranslate{page}\nobreakspace}% 
 \acrotranslate{pages}\nobreakspace}% 
 \tabularnewline% 
}%;% 
}{}
{\AcroRerun}% 
\acroheading% 
\acropreamble% 
{\acrotranslate{page}\nobreakspace}% 
{\acrotranslate{pages}\nobreakspace}% 
\tabularnewline
\}%;% 
}\acroRerun% 
{\acroheading% 
\acropreamble% 
}{\begin{tabular}{>{\bfseries}lp{.7\linewidth}}% 
\AcronymTable% 
{\end{tabular}% 
}
\NewAcroTemplate[list]{toc}{% 
\acroheading% 
\acropreamble% 
}{\begin{tabular}{c}{\bfseries l}% 
\AcronymTable% 
{\end{tabular}% 
}
\NewAcroTemplate[list]{toc}{% 
\acroheading% 
\acropreamble% 
}{\begin{tabular}{>{\bfseries}lp{.7\linewidth}}% 
\AcronymTable% 
{\end{tabular}% 
}
\end{verbatim}
25.6. New heading templates

Let’s take a look at the two templates `section` and `section*` which should give you enough information to build your own:

\begin{verbatim}
\NewAcroTemplate[heading]{section} {\section {\acrolistname}}
\NewAcroTemplate[heading]{section*}{\section*{\acrolistname}}
\end{verbatim}

26. Endings

Referring to section 13.2 on page 23 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:

\begin{verbatim}
\DeclareAcroEnding{plural}{s}{s}
\end{verbatim}

In general \DeclareAcroEnding{(foo)}{(x)}{(y)} defines these options

\begin{verbatim}
short-\langle foo\rangle-ending = \langle value\rangle \hspace{1cm} Initial: \langle x\rangle
long-\langle foo\rangle-ending = \langle value\rangle \hspace{1cm} Initial: \langle y\rangle
\end{verbatim}

and these properties

\begin{verbatim}
short-\langle foo\rangle = \langle value\rangle \hspace{1cm} Initial: \langle x\rangle
short-\langle foo\rangle-form = \langle value\rangle \hspace{1cm} (initially empty)
alt-\langle foo\rangle = \langle value\rangle \hspace{1cm} Initial: \langle x\rangle
\end{verbatim}
In addition another command is defined which is meant to be used in template definitions.

\acro{\langle foo \rangle}

This command tells the template that the ending \langle foo \rangle should be used.

Section 30 on page 50 has an example of how this can be used to define a possessive ending and commands that make use of them like this:

\acro\{MP\} \quad \text{Member’s of Parliament (MP’s)}

\section{27. Articles}

Referring to section 14.2 on page 24 this section explains how to define and use additional articles.

\DeclareAcroArticle{\langle name \rangle}{\langle default \rangle}

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

\acro\{indefinite\}\{a\}

In general \acro\{\langle foo \rangle\}\{\langle x \rangle\} defines the option

\langle foo \rangle = \{\langle value \rangle\}

and these properties

\acro\{\langle foo \rangle\}\{\langle value \rangle\} = \{\langle value \rangle\}

(initially empty)

\acro\{\langle foo \rangle\}\{\langle value \rangle\}

(initially empty)
28. Translations

\acro{foo}
This command tells the template that the article \emph{(foo)} should be used.

Section 30 on page 50 has examples of how this can be used to define definite articles and commands that make use of them like this:

\begin{verbatim}
1 \acs{hadopi} \par
2 \acl{hadopi}
\end{verbatim}

l’HADOPI
la Haute Autorité pour la diffusion des œuvres et la protection des droits sur l’Internet

28. Translations

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

\DeclareAcroTranslation{\emph{(key)}}{\emph{(language-translation list)}}

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

\AddAcroTranslated{\emph{(key)}}{\emph{(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 \emph{pages} keyword:

\begin{verbatim}
1 \DeclareAcroTranslation{pages}{
2  Fallback = pp\abbrdot ,
3  English = pp\abbrdot ,
4  French = pp\abbrdot ,
5  German = S\abbrdot ,
6  Portuguese = pp\abbrdot 
7 }\end{verbatim}

Translations for a language could be added this way:

9. \acro already has the translations for Italian.

48
29. Properties

As you know from section 6 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*?!|>({name})

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*?!|>({name1})>({name2})

This newly declares property \langle name1 \rangle and makes it an alias of property \langle name2 \rangle. This means that \langle name1 \rangle gets the same value that \langle name2 \rangle has unless it is set explicitly. Property \langle name2 \rangle must exist.

\MakeAcroPropertyAlias({name1})>({name2})

This makes property \langle name1 \rangle and makes it an alias of property \langle name2 \rangle. Both properties must exist.
30. Own acronym commands

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

30. Own acronym commands

30.1. Background

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

\NewAcroCommand\{command\}{arg. spec.}\{code\}

This creates the new command (command) with the argument specification so(arg. spec.) and replacement text (code). There are significant differences to \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.

\RenewAcroCommand\{command\}{arg. spec.}\{code\}

Like \NewAcroCommand but redefines an existing command.

\UseAcroTemplate\{type\}\{name\}\{argument number\}\{arguments\}

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

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

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

Equivalent definitions would be:

\NewAcroCommand\ac\{m\}\{\UseAcroTemplate\{acronym\}\{first\}\{#1\}\}
\NewAcroCommand\ac\{m\}\{\UseAcroTemplate\{acronym\}\{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 \acrowrite.

\footnote{10. in the sense of an xparse command.}
30. Own acronym commands

\acrocite
Tells ACRo to output the citation.

\acrodonotuse
Tells ACRo to not count this as usage.

\acroplural
Use plural form.

\acroindefinite
Use indefinite article

\acroupper
Use uppercase form.

\acrofull
Use first or full form.

Here is an example that makes use of them:

\begin{verbatim}
\NewAcroCommand\Iacs{m}{\acroupper\acroindefinite\UseAcroTemplate{short}{#1}}
\end{verbatim}

30.2. Create commands for possessive endings

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

\begin{verbatim}
\DeclareAcroEnding{possessive}{'s}{'s}
\end{verbatim}

Then you define commands which make use of this ending:

\begin{verbatim}
\NewAcroCommand\acg{m}{\acrofull\acropossessive\UseAcroTemplate{first}{#1}}
\NewAcroCommand\acsg{m}{\acropossessive\UseAcroTemplate{short}{#1}}
\NewAcroCommand\aclg{m}{\acropossessive\UseAcroTemplate{long}{#1}}
\NewAcroCommand\acfg{m}{\acroindefinite\acropossessive\UseAcroTemplate{first}{#1}}
\NewAcroCommand\iacsg{m}{\acropossessive\UseAcroTemplate{short}{#1}}
\end{verbatim}
You maybe also define acronyms with corresponding properties:\footnote{Bear with me if this is incorrect: English is not my native language.}:

\begin{verbatim}
\DeclareAcronym{MP}{
  short = MP ,
  long  = Member of Parliament,
  plural-form = Members of Parliament,
  long-possessive-form = Member's of Parliament
}
\end{verbatim}

Now you can use it like this:

\begin{verbatim}
This is the \acg{MP} first day at work after \dots
\end{verbatim}

\begin{verbatim}
This is the MP's first day at work after \ldots
\end{verbatim}

\section{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 \texttt{input} it.
2. Put in in a .sty file and include it with \texttt{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 \texttt{acsetup} using the option

\begin{verbatim}
load-style = {\langle name\rangle}
\end{verbatim}

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*{⟨style⟩}[⟨details⟩]

has an optional star which switches to expl3 syntax. It also has an optional argument ⟨details⟩
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]% 
}
Part IV.

Appendix

A. Examples

Example 1: Basic usage

Links: [\TeX] [PDF]

File: acro.example.basic.tex

\acsetup{
  make-links ,
  pages / display = first ,
  pages / fill = {, }
}
\DeclareAcronym{CDMA}{

1 Intro

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 discussion whether Code Division Multiple Access (CDMA) should have been chosen over TDMA.

Furthermore

The reader could have forgotten all the nice acronyms, so we repeat the meaning again.

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.

Just to mention it: There is another Used Acronym (UA), just for testing purposes!

Figure

Figure 1: A float also admits references like GSM or Code Division Multiple Access (CDMA).

2.1 Some chemistry and physics

Nicotinamide Adenine Dinucleotide (NAD$^+$) is a major electron acceptor in the oxidation of fuel molecules. The reactive part of NAD$^+$ is its nictinamide ring, a pyridine derivate.

One mol consists of $N_A$ atoms or molecules. There is a relation between the constant of Boltzmann and the Number of Avogadro:

$$k = \frac{R}{N_A} \quad (1)$$

Liquid Oxygen/Liquid Hydrogen (LOX/LH$_2$)

2.2 Some testing fundamentals

When testing Integrated Circuits (ICs), one typically wants to identify functional blocks to be tested separately. The latter are commonly indicated as Blocks Under Test (BUTs). To test a BUT requires defining a testing strategy...

Acronyms

LH$_2$ Liquid Hydrogen, 1

Example 2: Re-implement \acflike

Links: [\TeX] [PDF]

File: acro.example.acflike.tex

\RenewAcroCommand\acflike{mm}{%
  \acroformat{long}{#2} (\UseAcroTemplate{short}{#1})%
}
\begin{document}
\ac{cd} \par
\acflike{cd}{Rohling}

Compact Disc (CD)
Rohling (CD)
A. Examples

Example 3: Invisible command for backref
Links: [TeX] [PDF] [github]  File: acro.example.issue-109.tex

\begin{verbatim}
7 pages/seq=false
8 }
9 \DeclareAcronym{ny}{
10 short = NY ,
11 long = New York ,
12 }
\end{verbatim}

Example 4: Defining a definite article
Links: [TeX] [PDF] [github]  File: acro.example.issue-111.tex

\begin{verbatim}
7 \NewAcroCommand\dacs{m}{\acrodefinite
8 \UseAcroTemplate{short}{#1}}
9 \NewAcroCommand\Dacs{m}{\acroupper\acrodefinite
10 \UseAcroTemplate{short}{#1}}
11 \NewAcroCommand\dacl{m}{\acrodefinite
12 \UseAcroTemplate{long}{#1}}
13 \NewAcroCommand\Dacl{m}{\acroupper\acrodefinite
14 \UseAcroTemplate{long}{#1}}
15 \DeclareAcronym{hadopi}{
16 short = HADOPI ,
17 long = Haute Autorité pour la diffusion des Œuvres et la Protection des droits sur Internet
18 }
\end{verbatim}

Example 5: Write the list of acronyms to an external file
Links: [TeX] [PDF] [github]  File: acro.example.issue-119.tex

\begin{verbatim}
7 \immediate\openout\acro@list\jobname.ac\relax
8 \immediate\write\acro@list{\string\begin{description}}
9 \let\item\relax
10 \acronymsmapF{
11 \immediate\write\acro@list{%
12 \space\space
13 \item[\acrofield{##1}{short}]}
14 \acrofield{##1}{list}%
\end{verbatim}
A. Examples

Example 6: Insert word between acronym and citation
Links: [TeX] [PDF] [github] File: acro.example.issue-154.tex

```latex
short = UC ,
long = Universal Composability ,
cite = xxx
}
\begin{document}
We use the Universal Composability (UC) model 
\end{document}
```

Example 7: How to define a possessive ending
Links: [TeX] [PDF] File: acro.example.possessive.tex

```latex
\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

```latex
\RenewAcroTemplate[list]{description}{%
\acroheading
\acropreamble
\begin{description}
\acroynmsmapF{
\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 
    (% 
    \acrowrite{short} \acroifT{foreign}{, }\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: \textit{Escherichia coli}

Second use: \textit{E. coli}

Example 11: Capitalization

\begin{verbatim}
\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 azide-alkyne cycloaddition
B. Acronyms

Below all abbreviations are listed which have been defined for the manual. The ones without page numbers have been defined but haven’t been used.

CD  compact disc ................................................................. 5, 14

CTAN  Comprehensive \TeX\ Archive Network

e.g.  for example (Latin: exempli gratia) ........................................ 26

ECU  Steuergerät (Englisch: Electronic Control Unit) .......................... 25

etc.  et cetera ........................................................................... 15, 30

HADOPI  Haute Autorité pour la diffusion des œuvres et la protection des droits sur l’Internet 48

ID  identification string ............................................................. 7, 43, 50

JPEG/JPG  Joint Photographic Experts Group ................................. 13

LA  Los Angeles

LPPL  \LaTeX\ Project Public License ............................................ 3

MP  Member of Parliament ......................................................... 23, 47, 52

NATO  Organisation des Nordatlantikvertrags (Englisch: North Atlantic Treaty Organization)

NY  New York ........................................................................... 27

PDF  Portable Document Format .................................................. 10, 21, 33 f.

PNG  Portable Network Graphics ................................................ 16
C. References

PU Polyurethan ................................................................. 30
SW Sammelwerk ............................................................... 23
\TeX{}sx \TeX{} StackExchange
UFO unidentified flying object (German: unbekanntes Flugobjekt) ............. 12 f., 23, 26

C. References

url: https://www.ctan.org/pkg/supertabular/.

babel. version 3.70, Jan. 26, 2022 (or newer).
url: https://www.ctan.org/pkg/babel/.

url: https://www.ctan.org/pkg/longtable/.

polyglossia. version 1.53, Apr. 12, 2021 (or newer).
url: https://www.ctan.org/pkg/polyglossia/.

[Kle18] Josef Kleber. pdfcomment. version 2.4a, Nov. 1, 2018 (or newer).
url: https://www.ctan.org/pkg/pdfcomment/.

[L3P] The \LaTeX{}3 Project Team. xparse. Jan. 12, 2022 (or newer).
url: https://www.ctan.org/pkg/xparse/.

[Leh+22] Philipp Lehman et al., current maintainers: Philip Kime and Moritz Wemheuer.
biblatex. version 3.17, Feb. 2, 2022 (or newer).
url: https://www.ctan.org/pkg/biblatex/.

url: https://www.ctan.org/pkg/tabularray/.

[Nie22] Clemens Niederberger. translations. version 1.12, Feb. 5, 2022 (or newer).
url: https://www.ctan.org/pkg/translations/.

[Obe19] Heiko Oberdiek. accsupp. version 0.6, Dec. 5, 2019 (or newer).
url: https://www.ctan.org/pkg/accsupp/.

[Obe20] Heiko Oberdiek. bookmark. version 1.29, Nov. 6, 2020 (or newer).
url: https://www.ctan.org/pkg/bookmark/.

[ORT21] Heiko Oberdiek, Sebastian Rahtz, and The \LaTeX{}3 Project Team.
hyperref. version 7.00m, June 7, 2021 (or newer).
url: https://www.ctan.org/pkg/hyperref/.

url: https://www.ctan.org/pkg/glossaries/.
D. Index

| Symbols | \acrogroupcite | 42, 44 | \acroheading | 37, 44 f., 56 | \acs | 4 f., 12 |
|---------|----------------|--------|---------------|----------------|---------|
| \* (symbol) | \acroheading | 37, 44 f., 56 | \acs | 4 f., 12 |
| - (symbol) | 30 | \acs | 4 f., 17 ff., 21, 25, 28, 31, 52, 54 |
| . (symbol) | 30 f. | \acs | 4 f., 17 ff., 21, 25, 28, 31, 52, 54 |
| > (symbol) | 49 | \acs | 4 f., 17 ff., 21, 25, 28, 31, 52, 54 |
| ? (symbol) | 49 | \acs | 4 f., 17 ff., 21, 25, 28, 31, 52, 54 |

A

\abbrdot | 31 f., 48 f. | \addchap (template) | 39 |
\Ac | 4 f., 11 | \addchap+ (template) | 39 |
\ac | 4 f., 11–16, 23, 25, 27–30, 36 f., 50, 54 ff. | \addsec (template) | 40 |
\Aca | 12 | \addsec+ (template) | 40 |
\aca | 12 f. | \all | 26 |
\Acap | 12 | \alt | 20 |
\acap | 12 | \alt (property) | 7, 10, 13, 27, 34 |
\acbarrier | 19, 29 | \alt (template) | 38 |
\accssupp (package) | 10, 34 f. | \alt-ac (property) | 10, 34 |
\acdor | 14 f., 25, 30 ff. | \alt-format (property) | 9, 20 |
\Acf | 5, 12 | \alt-in (property) | 9, 20 |
\acf | 5, 12, 14, 16, 21, 25, 28 ff. | \alt-plural (property) | 8, 22 |
\acfg | 47, 51, 56 | \alt-plural-form (property) | 8, 22 |
\acflike | 54 | \acsetup | 39 |
\acfp | 12 | \acs | 4 f., 12 |
\acfrailing | 31 f. | \acs | 4 f., 12 |
\Acl | 4, 12 | \acs | 4 f., 12 |
\acl | 4 f., 12 | \accsupp (package) | 11, 24 f., 35 |
\acls | 51, 56 | \barriers | 29 |
\aclp | 12, 26 | \before-citation (property) | 8 |
\aclp | 12 | \BeginAccSupp | 34 |
\acp | 11 f. | \bezos, Javier | 11 |
\acp | 11 f. | \biblatex (package) | 28 |
\acreset | 23, 32 | \bookmark (package) | 10, 33 |
\acresetall | 32 | \braams, Johannes | 11, 39 |
\AcroAcronymsMap | 42 | \caption | 37 |
\AcroAddRow | 43, 45 | \caption (package) | 36 |
\acrocite | 51 | \carlisle, david | 39 |
\acrodef | 55, 58 | \case-insensitive | 7 |
\acrodefonuse | 51 | \casesensitive | 7 |
\acrodotfill | 18 | \CD | 5, 14 |
\acroendfootnote | 38 | \chapter (template) | 40 |
### Index

<table>
<thead>
<tr>
<th>chapter* (template)</th>
<th>40</th>
</tr>
</thead>
<tbody>
<tr>
<td>Charette, François</td>
<td>11</td>
</tr>
<tr>
<td>cite (property)</td>
<td>8, 27</td>
</tr>
<tr>
<td>clear</td>
<td>28</td>
</tr>
<tr>
<td>colspec</td>
<td>26 ff., 33</td>
</tr>
<tr>
<td>foreign-format (property)</td>
<td>9, 21, 24</td>
</tr>
<tr>
<td>foreign-local (property)</td>
<td>11, 24</td>
</tr>
<tr>
<td>foreign-plural (property)</td>
<td>9, 22, 24</td>
</tr>
<tr>
<td>foreign-plural-form (property)</td>
<td>9, 22, 24</td>
</tr>
<tr>
<td>format</td>
<td>41</td>
</tr>
<tr>
<td>format (property)</td>
<td>9, 20</td>
</tr>
<tr>
<td>group</td>
<td>27</td>
</tr>
<tr>
<td>HADOPI</td>
<td>48</td>
</tr>
<tr>
<td>heading</td>
<td>16</td>
</tr>
<tr>
<td>hyperref (package)</td>
<td>10, 17, 32 ff.</td>
</tr>
<tr>
<td>I</td>
<td>12, 23, 26</td>
</tr>
<tr>
<td>Iac</td>
<td>11 f., 23</td>
</tr>
<tr>
<td>Iac</td>
<td>11 f., 23</td>
</tr>
<tr>
<td>Iaca</td>
<td>12, 23</td>
</tr>
<tr>
<td>Iacs</td>
<td>12, 23, 51</td>
</tr>
<tr>
<td>extra</td>
<td>20</td>
</tr>
<tr>
<td>extra (property)</td>
<td>7, 10, 35</td>
</tr>
<tr>
<td>extra-ac (property)</td>
<td>10, 35</td>
</tr>
<tr>
<td>extra-format (property)</td>
<td>9, 20</td>
</tr>
<tr>
<td>fill</td>
<td>17, 43</td>
</tr>
<tr>
<td>first</td>
<td>26</td>
</tr>
<tr>
<td>first (template)</td>
<td>38</td>
</tr>
<tr>
<td>first-long</td>
<td>20</td>
</tr>
<tr>
<td>first-long-format (property)</td>
<td>9, 20</td>
</tr>
<tr>
<td>float</td>
<td>36</td>
</tr>
<tr>
<td>footnote (template)</td>
<td>38</td>
</tr>
<tr>
<td>foreign</td>
<td>20</td>
</tr>
<tr>
<td>foreign (property)</td>
<td>7, 10, 24, 35, 49</td>
</tr>
<tr>
<td>foreign-ac (property)</td>
<td>10, 35</td>
</tr>
<tr>
<td>foreign-babel (property)</td>
<td>11, 24 f.</td>
</tr>
<tr>
<td>foreign-format (property)</td>
<td>9, 21</td>
</tr>
<tr>
<td>foreign-local (property)</td>
<td>11, 24</td>
</tr>
<tr>
<td>foreign-plural (property)</td>
<td>9, 22, 24</td>
</tr>
<tr>
<td>foreign-plural-form (property)</td>
<td>9, 22, 24</td>
</tr>
<tr>
<td>format</td>
<td>41</td>
</tr>
<tr>
<td>format (property)</td>
<td>9, 20</td>
</tr>
<tr>
<td>group</td>
<td>27</td>
</tr>
<tr>
<td>HADOPI</td>
<td>48</td>
</tr>
<tr>
<td>heading</td>
<td>16</td>
</tr>
<tr>
<td>hyperref (package)</td>
<td>10, 17, 32 ff.</td>
</tr>
<tr>
<td>I</td>
<td>12, 23, 26</td>
</tr>
<tr>
<td>Iac</td>
<td>11 f., 23</td>
</tr>
<tr>
<td>Iac</td>
<td>11 f., 23</td>
</tr>
<tr>
<td>Iaca</td>
<td>12, 23</td>
</tr>
<tr>
<td>Iacs</td>
<td>12, 23, 51</td>
</tr>
<tr>
<td>Iacs</td>
<td>12, 23, 51</td>
</tr>
<tr>
<td>include</td>
<td>16, 18 f.</td>
</tr>
<tr>
<td>index</td>
<td>8, 29</td>
</tr>
<tr>
<td>index (property)</td>
<td>8, 28 f.</td>
</tr>
<tr>
<td>index-cmd (property)</td>
<td>8</td>
</tr>
<tr>
<td>index-sort (property)</td>
<td>8, 29</td>
</tr>
<tr>
<td>JPEG</td>
<td>13</td>
</tr>
<tr>
<td>Jurriens, Theo</td>
<td>39</td>
</tr>
<tr>
<td>K</td>
<td>28</td>
</tr>
<tr>
<td>Kime, Philip</td>
<td>28</td>
</tr>
<tr>
<td>Kleeber, Josef</td>
<td>33</td>
</tr>
<tr>
<td>L</td>
<td>35</td>
</tr>
<tr>
<td>language</td>
<td>35</td>
</tr>
<tr>
<td>Lehman, Philipp</td>
<td>28</td>
</tr>
<tr>
<td>link-only-first</td>
<td>32</td>
</tr>
<tr>
<td>list</td>
<td>3, 17, 41, 49</td>
</tr>
<tr>
<td>list</td>
<td>20</td>
</tr>
<tr>
<td>list (property)</td>
<td>10, 35</td>
</tr>
<tr>
<td>list (property)</td>
<td>10, 35</td>
</tr>
<tr>
<td>list-ac (property)</td>
<td>10, 35</td>
</tr>
<tr>
<td>list-format (property)</td>
<td>9, 21</td>
</tr>
<tr>
<td>list/display</td>
<td>16</td>
</tr>
<tr>
<td>lists</td>
<td>37</td>
</tr>
<tr>
<td>load-style</td>
<td>52</td>
</tr>
<tr>
<td>local</td>
<td>19</td>
</tr>
<tr>
<td>locale</td>
<td>11, 24</td>
</tr>
<tr>
<td>lof (template)</td>
<td>39, 43</td>
</tr>
<tr>
<td>long</td>
<td>20</td>
</tr>
<tr>
<td>long (property)</td>
<td>4, 7, 10, 34</td>
</tr>
<tr>
<td>long (template)</td>
<td>38</td>
</tr>
<tr>
<td>long-ac (property)</td>
<td>10, 34 f.</td>
</tr>
<tr>
<td>long-format (property)</td>
<td>9, 20 f.</td>
</tr>
<tr>
<td>long-indefinite (property)</td>
<td>9</td>
</tr>
<tr>
<td>long-plural (property)</td>
<td>8, 22</td>
</tr>
<tr>
<td>long-plural-ending</td>
<td>22</td>
</tr>
<tr>
<td>long-plural-form (property)</td>
<td>8, 22</td>
</tr>
<tr>
<td>long-post (property)</td>
<td>7</td>
</tr>
<tr>
<td>long-short (template)</td>
<td>38</td>
</tr>
<tr>
<td>longtable</td>
<td>37</td>
</tr>
<tr>
<td>longtable (template)</td>
<td>39</td>
</tr>
<tr>
<td>longtable (package)</td>
<td>39</td>
</tr>
<tr>
<td>LPFL</td>
<td>37</td>
</tr>
<tr>
<td>ltxttable</td>
<td>37</td>
</tr>
<tr>
<td>LYU, Jianrui</td>
<td>39</td>
</tr>
<tr>
<td>M</td>
<td>19, 32</td>
</tr>
<tr>
<td>make-links</td>
<td>19, 32</td>
</tr>
<tr>
<td>MakeAcroPropertyAlias</td>
<td>49</td>
</tr>
<tr>
<td>maketitle</td>
<td>37</td>
</tr>
<tr>
<td>method</td>
<td>34</td>
</tr>
<tr>
<td>mfirstuc (package)</td>
<td>27</td>
</tr>
<tr>
<td>MP</td>
<td>23, 47, 52</td>
</tr>
<tr>
<td>N</td>
<td>17 f., 41, 43</td>
</tr>
<tr>
<td>name</td>
<td>17 f., 41, 43</td>
</tr>
<tr>
<td>New York City</td>
<td>27</td>
</tr>
<tr>
<td>NewAcroCommand</td>
<td>50 f., 55 f., 58</td>
</tr>
<tr>
<td>NewAcroPreset</td>
<td>11</td>
</tr>
<tr>
<td>NewAcroTemplate</td>
<td>49, 43, 46</td>
</tr>
<tr>
<td>Niederberger, Clemens</td>
<td>35</td>
</tr>
<tr>
<td>no-index (property)</td>
<td>29</td>
</tr>
<tr>
<td>none (template)</td>
<td>40</td>
</tr>
<tr>
<td>NY</td>
<td>27</td>
</tr>
<tr>
<td>O</td>
<td>10, 32 ff.</td>
</tr>
<tr>
<td>Oberdiek, Heiko</td>
<td>10, 32 ff.</td>
</tr>
<tr>
<td>options</td>
<td>34</td>
</tr>
<tr>
<td>P</td>
<td>3, 42 f.</td>
</tr>
<tr>
<td>PDF</td>
<td>21</td>
</tr>
<tr>
<td>PDF</td>
<td>10, 21, 33 f.</td>
</tr>
<tr>
<td>pdfcomment (property)</td>
<td>10, 33</td>
</tr>
<tr>
<td>pdfcomment (package)</td>
<td>33</td>
</tr>
</tbody>
</table>
### Index

<table>
<thead>
<tr>
<th>Command/Package</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>pdfcomment/cmd</td>
<td>10</td>
</tr>
<tr>
<td>pdfcomment/use</td>
<td>10</td>
</tr>
<tr>
<td>pdfstring (property)</td>
<td>10, 33</td>
</tr>
<tr>
<td>PNG</td>
<td>16</td>
</tr>
<tr>
<td>polyglossia (package)</td>
<td>11, 24 f., 35</td>
</tr>
<tr>
<td>post (property)</td>
<td>7</td>
</tr>
<tr>
<td>pre</td>
<td>17, 27 f.</td>
</tr>
<tr>
<td>preamble</td>
<td>17</td>
</tr>
<tr>
<td>preset (property)</td>
<td>11</td>
</tr>
<tr>
<td>\printacronyms</td>
<td>16–19, 37, 53</td>
</tr>
<tr>
<td>PU</td>
<td>30</td>
</tr>
<tr>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Rahtz, Sebastian</td>
<td>10, 32 f.</td>
</tr>
<tr>
<td>\RenewAcroCommand</td>
<td>50, 54</td>
</tr>
<tr>
<td>\RenewAcroPreset</td>
<td>11</td>
</tr>
<tr>
<td>\RenewAcroTemplate</td>
<td>40, 56 f.</td>
</tr>
<tr>
<td>replace</td>
<td>21</td>
</tr>
<tr>
<td>reset</td>
<td>29</td>
</tr>
<tr>
<td>Reutenauer, Arthur</td>
<td>11</td>
</tr>
<tr>
<td>S</td>
<td></td>
</tr>
<tr>
<td>section (template)</td>
<td>40, 46</td>
</tr>
<tr>
<td>section* (template)</td>
<td>40, 46</td>
</tr>
<tr>
<td>\SetupAcroTemplate</td>
<td>40</td>
</tr>
<tr>
<td>\SetupNextAcroTemplate</td>
<td>40</td>
</tr>
<tr>
<td>short (property)</td>
<td>4, 7 f., 10, 27, 33 f.</td>
</tr>
<tr>
<td>short (template)</td>
<td>38</td>
</tr>
<tr>
<td>short-acc (property)</td>
<td>10, 34</td>
</tr>
<tr>
<td>short-format (property)</td>
<td>9, 20</td>
</tr>
<tr>
<td>short-indefinite (property)</td>
<td>9</td>
</tr>
<tr>
<td>short-long (template)</td>
<td>38</td>
</tr>
<tr>
<td>short-plural (property)</td>
<td>8, 22</td>
</tr>
<tr>
<td>short-plural-ending</td>
<td>22</td>
</tr>
<tr>
<td>short-plural-form (property)</td>
<td>8, 22</td>
</tr>
<tr>
<td>show (template)</td>
<td>39</td>
</tr>
<tr>
<td>single</td>
<td>7, 15, 29, 39</td>
</tr>
<tr>
<td>single (property)</td>
<td>7, 15, 39</td>
</tr>
<tr>
<td>single (template)</td>
<td>15, 39</td>
</tr>
<tr>
<td>single-acc (property)</td>
<td>10, 35</td>
</tr>
<tr>
<td>single-format (property)</td>
<td>21</td>
</tr>
<tr>
<td>single-style</td>
<td>15</td>
</tr>
<tr>
<td>single-style (property)</td>
<td>10, 21</td>
</tr>
<tr>
<td>sort</td>
<td>16</td>
</tr>
<tr>
<td>sort (property)</td>
<td>8, 29</td>
</tr>
<tr>
<td>subsequent-style (property)</td>
<td>14</td>
</tr>
<tr>
<td>subsequent-style (property)</td>
<td>9</td>
</tr>
<tr>
<td>supertabular (template)</td>
<td>39</td>
</tr>
<tr>
<td>supertabular (package)</td>
<td>39</td>
</tr>
<tr>
<td>SW</td>
<td>23</td>
</tr>
<tr>
<td>T</td>
<td></td>
</tr>
<tr>
<td>tabu</td>
<td>37</td>
</tr>
<tr>
<td>tabular (template)</td>
<td>39, 43 f.</td>
</tr>
<tr>
<td>tabulararray (template)</td>
<td>39</td>
</tr>
<tr>
<td>tabulararray (package)</td>
<td>39</td>
</tr>
<tr>
<td>tabularx</td>
<td>37</td>
</tr>
<tr>
<td>tag (property)</td>
<td>8, 18</td>
</tr>
<tr>
<td>Talbot, Nicola L.C.</td>
<td>27</td>
</tr>
<tr>
<td>template</td>
<td>16</td>
</tr>
<tr>
<td>The L\TeX3 Project Team</td>
<td>10, 32 f., 50</td>
</tr>
<tr>
<td>threshold</td>
<td>17</td>
</tr>
<tr>
<td>title</td>
<td>26</td>
</tr>
<tr>
<td>toc (template)</td>
<td>39, 42 ff.</td>
</tr>
<tr>
<td>translations (package)</td>
<td>35</td>
</tr>
<tr>
<td>U</td>
<td></td>
</tr>
<tr>
<td>UFO</td>
<td>12 f., 23, 26</td>
</tr>
<tr>
<td>upgrade</td>
<td>5</td>
</tr>
<tr>
<td>use</td>
<td>17, 19, 28 f., 33 f.</td>
</tr>
<tr>
<td>use-id-as-short</td>
<td>7</td>
</tr>
<tr>
<td>\UseAcroTemplate</td>
<td>50 f., 54 ff., 58 f.</td>
</tr>
<tr>
<td>uselist (property)</td>
<td>11</td>
</tr>
<tr>
<td>W</td>
<td></td>
</tr>
<tr>
<td>Wemheuer, Moritz</td>
<td>28</td>
</tr>
<tr>
<td>Wikipedia</td>
<td>27</td>
</tr>
<tr>
<td>X</td>
<td></td>
</tr>
<tr>
<td>xparse (package)</td>
<td>50</td>
</tr>
</tbody>
</table>