ACRO

v2.11d 2020/03/07

Typeset Acronyms and other Abbreviations

Clemens Niederberger

https://bitbucket.org/cgnieder/acro/

contact@mychemistry.eu

ACRO allows you to define and use abbreviations in a simple way. Abbreviations can be divided into different classes of abbreviations. Lists of abbreviations can be created (also of separate classes of abbreviations) and printed wherever you want the lists to appear.

ACRO provides an option single which ignores abbreviations that are used only once in the whole document.

As an experimental feature ACRO also offers the option sort which automatically sorts the list created by \printacronyms.

ACRO also has the feature of creating local lists

Table of Contents

1. Licence and Requirements 2
   3.4. Fetching the Single Appearance 14
   3.5. Using Classes . . . . . . . . . 14
   3.6. Reset or Mark as Used, Test if Acronym Has Been Used . . 15
   3.7. \ac and Friends in PDF Bookmarks, Accessibility Support,
        Tooltips . . . . . . . . . . . . . 16
   3.7.1. PDF Bookmarks . . . . . 16
   3.7.2. Accessibility Support . 17
   3.7.3. Tooltips . . . . . . . . . 17
   3.8. Adding Acronyms to the Index 18

2. Basics 2
   2.1. Creating New Acronyms . . . 2
   2.2. Logging of Acronyms . . . . 6
   2.3. Using the Acronyms – the Commands . . . . . . . . . 7
   2.4. Plural Forms . . . . . . . . . 9
   2.5. Alternative Short Forms . . . . 10
   2.6. Extra Information for the List Entry . . . . . . . . . . . . 11
   2.7. Foreign Language Acronyms . 11

3. Additional Commands and Possibilities 12
   3.1. Indefinite Forms . . . . . . . . 12
   3.2. Uppercasing . . . . . . . . . 13
   3.3. Simulating the First Appearance 13

4. Printing the List 18

5. Options and Customization 19
   5.1. General Options . . . . . . . . . 19
   5.2. Options Regarding Acronyms 22
   5.3. Options Regarding the List . . 24
1. Licence and Requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the \LaTeX\ Project Public License (\LPPL), version 1.3 or later (http://www.latex-project.org/lppl.txt). The software has the status “maintained.” \acro loads and needs the following packages: expl3, xparse, xtemplate, l3keys2e, zref-abspage and translations [Nie17].

2. Basics

2.1. Creating New Acronyms

Acronyms are created with the command \texttt{\DeclareAcronym}.\acro

\texttt{\DeclareAcronym\{\texttt{id}\}\{\texttt{list of keys}\}}

The basic command for declaring an acronym where \texttt{id} is a unique string identifying the acronym. Per default behaviour this is case sensitive which means \texttt{id} is different from \texttt{ID}, for example. There is an option \texttt{case-sensitive} to change this.

This command understands a number of keys which are listed below. Some of them are not described immediately but at appropriate places in the documentation.

\texttt{short = \{\texttt{text}\}\}}\texttt{\}}\texttt{(required)}

The short form of the acronym. \textit{This property is required}: an acronym must have a short form.

---

1. \texttt{on ctan as l3kernel}: http://mirrors.ctan.org/macros/latex/contrib/l3kernel/
2. \texttt{on ctan as l3packages}: http://mirrors.ctan.org/macros/latex/contrib/l3packages/
3. \texttt{on ctan as oberdiek}: http://mirrors.ctan.org/macros/latex/contrib/oberdiek/
4. \texttt{on ctan as translations}: http://mirrors.ctan.org/macros/latex/contrib/translations/
If this is set it must be set as first option! If another property is set first and notices the short property missing it assumes that the ID should be used as short version and sets it accordingly. A warning will be written to the log then.

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

short-plural = \{(text)\}  
The plural ending appended to the short form.

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

long-plural = \{(text)\}  
The plural ending appended to the long form.

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

alt-plural = \{(text)\}  
The plural ending appended to the alternative form.

alt-plural-form = \{(text)\}  
The plural alternative form of the acronym; replaces the alternative form when used instead of appending the plural ending.

foreign-plural = \{(text)\}  
The plural ending appended to the foreign form.

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

list = \{(text)\}  
If specified this will be written in the list as description instead of the long form.

short-indefinite = \{(text)\}  
Indefinite article for the short form.

long-indefinite = \{(text)\}  
Indefinite article for the long form.

long-pre = \{(text)\}  
\{(text)\} is prepended to the long form in the text but not in the list of acronyms.

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


2. Basics

\texttt{post} = \{(text)\}

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

\texttt{alt} = \{(text)\}

Alternative short form.

\texttt{alt\text{-}indefinite} = \{(text)\}

Indefinite article for the alternative form.

\texttt{extra} = \{(text)\}

Extra information to be added in the list of acronyms.

\texttt{foreign} = \{(original long form)\}

Can be useful when dealing with acronyms in foreign languages, see section 2.7 on page 11 for details.

\texttt{foreign\text{-}lang} = \{(language)\}

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

\texttt{single} = \{(text)\}

If provided \texttt{\{(text)\}} will be used instead of the long form if the acronym is only used a single time and the option \texttt{\texttt{single} = {true}} is active.

\texttt{sort} = \{(text)\}

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

\texttt{class} = \{\texttt{csv list}\}

The class(es) the acronym belongs to.

\texttt{cite} = [\{(prenote)\}][\{(postnote)\}][\{(citation keys)\}]

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

\texttt{before\text{-}citation} = \{(code)\}

\texttt{(code)} is inserted directly before \texttt{cite} and only when \texttt{cite} is used.

\texttt{after\text{-}citation} = \{(code)\}

\texttt{(code)} is inserted directly after \texttt{cite} and only when \texttt{cite} is used.

\texttt{short\text{-}format} = \{(\LaTeX\ code)\}

The format used for the short form of the acronym.

\texttt{long\text{-}format} = \{(\LaTeX\ code)\}

The format used for the long form of the acronym.

\texttt{alt\text{-}format} = \{(\LaTeX\ code)\}

The format used for the alternative form of the acronym. If this is not given the short format will be used.
In its simplest form an acronym needs a short and a long form. Please note that both keys must be set and that the short property must always be the first property that is set.

```latex
\% preamble:
\DeclareAcronym{test}{
  short = ST ,
  long = Some Test
}
```
2. Basics

This creates the acronym “ST” with the i.d “test” and the long form “Some Test.”

The short-format key allows you to choose a specific format for the short form of an acronym:

```
% preamble:
\DeclareAcronym{ot}{
  short   = ot ,
  long    = Other Test ,
  short-format = \scshape
}
```

The short form now looks like this: ot.

The cite key needs a bit explaining. It expects arguments like the standard \cite command, i.e., two optional arguments setting the ⟨prenote⟩ and ⟨postnote⟩ and one mandatory argument setting the citation key.

```
% preamble:
\DeclareAcronym{ny}{
\DeclareAcronym{ny}{
  short   = NY ,
  short-plural = ,
  long    = New York ,
  long-plural = ,
  cite    = \{NewYork\}
}
```

```
% bib file for use with biber/biblatex:
@online{NewYork,
  author = {Wikipedia},
  title  = {New York City},
  urldate = {2012-09-27},
  url     = {http://en.wikipedia.org/wiki/New_York_City},
  year    = {2012}
}
```

The first appearance now looks as follows\(^5\): New York (NY) [Wik12].

2.2. Logging of Acronyms

When you activate \acro’s option \log \acro` writes information about the acronyms it defines to the log file.

---

\(^5\) The appearance of the citation of course depends on the citation style you’re using.
2. Basics

\[\text{log} = \text{true}|\text{false}|\text{silent}|\text{verbose}\]  Default: false

When set to true/silent ACRo writes the main properties of an acronym to the log file. When set to verbose ACRo writes all properties of an acronym to the log file.

This is an example of the logging info with \text{log} = \{true\} or \text{log} = \{silent\}.

\begin{verbatim}
| Package acro Info -- defining new acronym: |
| ID = \{jpg\} |
| short = \{JPEG\} |
| long = \{Joint Photographic Experts Group\} |
| alt = \{JPG\} |
| sort = \{jpeg\} |
| class = {} |
| list = {} |
| extra = {} |
| foreign = {} |
| pdfstring = {} |
| cite = {} |
\end{verbatim}

2.3. Using the Acronyms – the Commands

Acronyms are used with one of the following commands:

\[\text{\textbackslash ac*}\{\langle id\rangle}\]

basic command; the first output is different from subsequent ones.

\[\text{\textbackslash Ac*}\{\langle id\rangle}\]

same as \text{\textbackslash ac} but capitalizes the first letter. Obeys the option uppercase-short.

\[\text{\textbackslash acs*}\{\langle id\rangle}\]

short form; the actual acronym.

\[\text{\textbackslash Acs*}\{\langle id\rangle}\]

same as \text{\textbackslash acs} but capitalizes the first letter.

\[\text{\textbackslash acl*}\{\langle id\rangle}\]

long form; the meaning of the acronym.

\[\text{\textbackslash Acl*}\{\langle id\rangle}\]

same as \text{\textbackslash acl} but capitalizes first letter.

\[\text{\textbackslash aca*}\{\langle id\rangle}\]

alternative short form as specified in the alt property of \texttt{\textbackslash DeclareAcronym}; if it hasn’t been specified this is identical to \texttt{\textbackslash acs}.
2. Basics

\texttt{\textbackslash aca}\{\langle id\rangle\}

same as \texttt{\textbackslash aca} but capitalizes the first letter.

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

first form; output like the first time \texttt{\textbackslash ac} is output.

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

same as \texttt{\textbackslash acf} but capitalizes first letter. Obeys the option \texttt{uppercase-short}.

\texttt{\textbackslash acp}\{\langle id\rangle\}

plural form of \texttt{\textbackslash ac};

\texttt{\textbackslash acp}\{\langle id\rangle\}

same as \texttt{\textbackslash acp} but capitalizes first letter. Obeys the option \texttt{uppercase-short}.

\texttt{\textbackslash acsp}\{\langle id\rangle\}

plural form of \texttt{\textbackslash acs};

\texttt{\textbackslash acsp}\{\langle id\rangle\}

same as \texttt{\textbackslash acsp} but capitalizes the first letter.

\texttt{\textbackslash acl}\{\langle id\rangle\}

plural form of \texttt{\textbackslash acl};

\texttt{\textbackslash acl}\{\langle id\rangle\}

same as \texttt{\textbackslash acl} but capitalizes first letter.

\texttt{\textbackslash acap}\{\langle id\rangle\}

plural form of \texttt{\textbackslash aca};

\texttt{\textbackslash acap}\{\langle id\rangle\}

same as \texttt{\textbackslash acap} but capitalizes the first letter.

\texttt{\textbackslash acfp}\{\langle id\rangle\}

plural form of \texttt{\textbackslash acf};

\texttt{\textbackslash acfp}\{\langle id\rangle\}

same as \texttt{\textbackslash acfp} but capitalizes first letter of the long form. Obeys the option \texttt{uppercase-short}.

If an acronym is used the first time with \texttt{\textbackslash ac} its output is different from subsequent uses. To be clear on this: the first time! If the acronym has been used with \textit{any} of the output commands before it is \textit{not} the first time any more.

If you use the starred variant an acronym will not be marked as used. This proves useful if an acronym is typeset in a section title, for example, since then the appearance in the table of contents won’t mark it as used.
2. Basics

\begin{verbatim}
\% preamble:
\% \DeclareAcronym{cd}{
  \short = cd ,
  \long = Compact Disc ,
  \short-format = \scshape
}
\first time: Compact Disc (CD)
second time: CD
short: CD
alternative: cd
first again: Compact Disc (CD)
long: Compact Disc
short plural: CDs
long plural: Compact Discs
\end{verbatim}

2.4. Plural Forms

If an acronym is defined in the standard way \texttt{acro} uses an ‘s’ that’s appended to both the short and the long form when one of the plural commands is used. However, that is not always the best solution. For one thing not all acronyms may have a plural form. Second, the plural form especially of the long forms may be formed differently. And third, other languages can have other plural endings.

For these reasons \texttt{DeclareAcronym} can get the following keys:

- \texttt{short-plural} = \{\langle text\rangle\}
  Default: s
  The plural ending of the short form.

- \texttt{long-plural} = \{\langle text\rangle\}
  Default: s
  The plural ending of the long form.

- \texttt{long-plural-form} = \{\langle text\rangle\}
  An alternative plural form for the long form.

These keys are optional. If they’re not used, the default setting is \texttt{s}. If you use \texttt{long-plural-form} the long form will be replaced by the specified plural form when necessary.

Suppose we define the following acronyms:

\begin{verbatim}
\DeclareAcronym{cd}{
  \short = cd ,
  \long = Compact Disc ,
  \short-format = \scshape
}
\DeclareAcronym{ny}{
  \short = NY ,
  \short-plural = ,
\end{verbatim}
2. Basics

These acronyms now have the following plural appearances:

- \acs{cd}, \acl{cd} \quad CDs, Compact Discs
- \acs{ny}, \acl{ny} \quad NY, New York
- \acs{sw}, \acl{sw} \quad SWs, Sammelwerke
- \acs{MP}, \acl{MP} \quad MPs, Members of Parliament

In the default setting the format of the long or short form is not extended to the plural ending which might look strange if you choose bold weight as format, for example. In this case you can tell \acro to extend the format.

\texttt{format-include-endings = true|false}

Default: false

Tell \acro to use the format of the base form for the endings, too.

For a more general view on endings please read section 12 on page 33.

2.5. Alternative Short Forms

For some acronyms it might be useful to have alternative forms. For this \acro has another key:

\texttt{alt = \{⟨text⟩\}}

Alternative short form.
2. Basics

The alternative form uses the same plural ending as the default short form and is formatted in the same way.

2.6. Extra Information for the List Entry

Of course you can print a list of acronyms where their meaning is explained. Sometimes it can be useful to add additional information there. This is done with another key to \DeclareAcronym:

extra = \{⟨text⟩\}

Additional information for the list of acronyms.

These information will only be displayed in the list. See section 4 on page 18 for the impact of the following example.

The \ac{nato} is an intergovernmental military alliance based on the North Atlantic Treaty which was signed on 4 April 1949. \ac{nato} headquarters are in Brussels, Belgium, one of the 28 member states across North America and Europe, the newest of which, Albania and Croatia, joined in April 2009.

The North Atlantic Treaty Organization (NATO) is an intergovernmental military alliance based on the North Atlantic Treaty which was signed on 4 April 1949. NATO headquarters are in Brussels, Belgium, one of the 28 member states across North America and Europe, the newest of which, Albania and Croatia, joined in April 2009.

2.7. Foreign Language Acronyms

I repeatedly read the wish for being able to add translations to acronyms when the acronyms stem from another language than the document language, i.e., something like the following in a German document:
3. Additional Commands and Possibilities

That’s why I decided to add the following properties:

foreign = "original long form"
A description for an acronym originating in another language than the document language.

foreign-lang = "language"
The babel \texttt{Babel}\cite{Bra91} or polyglossia \texttt{Polyglossia}\cite{Cha91} language of the foreign form. This language is used to wrap the entry with \foreignlanguage{language}.

Here is the definition of the above mentioned ECU acronym:

\begin{verbatim}
\DeclareAcronym{ecu}{
  short = ECU,
  long = Steuergerät,\`at,
  foreign = Electronic Control Unit,
  foreign-lang = english
}
\end{verbatim}

As you have seen this adds the foreign entry to the first appearance of an acronym. It is also added in parentheses to the list of acronyms after the long entry. Actually the entry there is the argument to the following command:

\acroenparen{argument}
Places \texttt{example} in parentheses: \acroenparen{example}: (example). See page 23 for a way to customize this other than redefining it.

3. Additional Commands and Possibilities

3.1. Indefinite Forms

Unlike many other languages\footnote{Let’s better say: unlike the other languages where I know at least the basics.} in English the indefinite article is not determined by the grammatical case, gender or number but by the pronunciation of the following word. This means that the short and the long form of an acronym can have different indefinite articles. For these cases \acro offers the following keys:

\begin{verbatim}
short-indefinite = "article" \hspace{1cm} Default: a
alt-indefinite = "article" \hspace{1cm} Default: a
\end{verbatim}

Indefinite article for the short form.
Indefinite article for the alternative form.
3. Additional Commands and Possibilities

\texttt{long-indefinite} = \{\langle\text{article}\rangle\} \quad \text{Default: a}

Indefinite article for the long form.

For every lowercase singular command two alternatives exist, preceded by \texttt{i} and \texttt{I}, respectively, which output the lowercase and uppercase version of the corresponding indefinite article.

\begin{verbatim}
\% preamble:
\% \DeclareAcronym{ufo}{
\% short = UFO ,
\% long = unidentified flying object ,
\% long-indefinite = an
\% }
\% \Iac{ufo}; \iacs{ufo}; \iacl{ufo}
\end{verbatim}

An unidentified flying object (UFO); a UFO; an unidentified flying object

3.2. Uppercasing

\texttt{\acfirstupper}\{\langle\text{token list}\rangle\}

This command uppercases the first token in \langle\text{token list}\rangle. The command is less powerful than \texttt{\makefirstuc} that is provided by the mfirstuc package \cite{Tal17} but it is expandable. Obvious downsides are for example that it does not uppercase accented letters.

The following options control the behaviour of the uppercasing commands:

\texttt{uc-cmd} = \{\langle\text{control sequence}\rangle\} \quad \texttt{Default: \acfirstupper}

The command that is used to capitalize the first word in the \texttt{\Ac} and the like commands. You can change it to another one like for example \texttt{\makefirstuc} \cite{Tal17} or \texttt{\MakeTextUppercase} \cite{vch16}.

\texttt{uppercase-short} = \texttt{true}|\texttt{false} \quad \texttt{Default: true}

If set to \texttt{true} commands like \texttt{\Ac} or \texttt{\Acp} will uppercase the first letter of the short form. Commands like \texttt{\Acs} or \texttt{\Acsp} will uppercase the short form in any case.

3.3. Simulating the First Appearance

Users told me\footnote{Well – one, to be precise :) that there are cases when it might be useful to have the the acronym typeset according to the \texttt{first-style} option (or \texttt{first-style} property) but with another text than the long form. For such cases \texttt{ACRO} offers the following commands.

\texttt{\acflike*}\{\langle\text{id}\rangle\}\{\langle\text{instead of long form}\rangle\}

Write some alternative long form for acronym with \texttt{1D} \langle\text{id}\rangle as if it were the first time the acronym was used.

7. from the mfirstuc package
8. from the textcase package

\bibliographystyle{plain}
\bibliography{example}

13
3. Additional Commands and Possibilities

\acfplike*{⟨id⟩}{⟨instead of long form⟩}
Plural form of \acflike.

\begin{itemize}
\item \acssetup{first-style=footnote}
\item \acflike{ny}{the big apple}
\end{itemize}

NY\textsuperscript{a}
\begin{itemize}
\item a. the big apple [Wik12]
\end{itemize}

The plural ending in \acsflike is only appended to the short form. It makes no sense to append it to the text that is inserted manually anyway. Note that whatever text you’re inserting might be gobbled depending on the \texttt{first-style} you’re using.

### 3.4. Fetching the Single Appearance

There are macros that fetch the single appearance of an acronym even if it has been used more than once and the \texttt{single} option is active.

\acssingle*{⟨id⟩}
Write acronym as if it were used only a single time.

\acsingles*{⟨id⟩}
Uppercase form of \acssingle.

\acpssingle*{⟨id⟩}
Plural form of \acssingle.

\acssingle*{⟨id⟩}
Plural form of \acsingles.

\begin{itemize}
\item \acsingle{ny}
New York [Wik12]
\end{itemize}

### 3.5. Using Classes

The acronyms of ACRO can be divided into different classes. This doesn’t change the output but allows different acronym lists, see section 4 on page 18. For this \texttt{\DeclareAcronym} has an additional key:

\texttt{class} = \{⟨csv list⟩\}
Associated class(es) for an acronym.

This might be useful if you can and want to divide your acronyms into different types, technical and grammatical ones, say, that shall be listed in different lists. Since every acronym can get a list of associated classes those classes can effectively be used like tags for filtering acronyms.
3.6. Reset or Mark as Used, Test if Acronym Has Been Used

If you want for some reason to fool \textsc{acro} into thinking that an acronym is used for the first time you can call one of these commands:

\begin{verbatim}
\acsreset\{comma separated list of ids\}
\end{verbatim}

This will reset a used acronym such that the next use of \texttt{ac} will again print it as if it were used the first time. This will \textit{not} remove an acronym from being printed in the list if it actually \textit{has} been used before.

\begin{verbatim}
\acsresetall
\end{verbatim}

Reset all acronyms.

\begin{verbatim}
\acsifused\{id\}\{true\}\{false\}
\end{verbatim}

This command tests if the acronym with ID \texttt{id} has already been used and either puts \texttt{true} or \texttt{false} in the input stream.

\begin{verbatim}
\acsiffirst\{true\}\{false\}
\end{verbatim}

To be used inside of acronym definitions in order to test if we are inside the first instance of the acronym or not.

\begin{verbatim}
\acsreset\{ny\}\ac\{ny\}
\end{verbatim}

New York (NY) [Wik12]

Beware that both commands act \textit{globally}! There are also commands that effectively do the opposite of \texttt{acsreset}, \textit{i.e.}, mark acronyms as used:
This has the same effect as if an acronym had been used twice, that is, further uses of \ac will print the short form and the acronym will in any case be printed in the list (as long as its class is not excluded).

\acuse{(comma separated list of ids)}

Mark all acronyms as used.

Then there are two further commands related to using acronyms:

\acswitchoff

This command is for patching in certain situations. For example some table environments like \texttt{tabularx} or \texttt{tabu} pass their content two or more times for determining the width of the table columns. Those can be patched to add \acswitchoff to their trial phase.

\acswitchon

Effectively the opposite of \acswitchoff – this command should probably never be needed.

\section*{3.7. \texttt{ac} and Friends in \texttt{PDF} Bookmarks, Accessibility Support, Tooltips}

\subsection*{3.7.1. \texttt{PDF} Bookmarks}

\texttt{acro}’s commands usually are not expandable which means they’d leave unallowed tokens in \texttt{PDF} bookmarks. \texttt{hyperref} offers \texttt{\texordfs} to circumvent that issue manually but that isn’t really a nice solution. What’s the point of having macros to get output for you if you have to specify it manually after all?

That is why \texttt{acro} offers a preliminary solution for this. In a bookmark every \ac like command falls back to a simple text string typesetting what \acs would do (or \acsp for plural forms). These text strings both can accessed manually and can be modified to an output reserved for \texttt{PDF} bookmarks.

\acpdfstring{(id)}

Access the text string used in \texttt{PDF} bookmarks.

\acpdfstringplural{(id)}

Access the plural form of the text string used in \texttt{PDF} bookmarks.

\pdfstring = \{\texttt{pdfstring}/\{plural ending\}\}

Key for \texttt{\DeclareAcronym} to declare a custom text string for \texttt{PDF} bookmarks. The plural ending can be set optionally.

For example the \texttt{PDF} acronym used in the title for this section is defined as follows:

```
\DeclareAcronym{pdf}{
  short = pdf,
  long = Portable Document Format,
}
3. Additional Commands and Possibilities

```latex
\begin{verbatim}
4  format = \scshape ,
5  pdfstring = PDF ,
6  accsupp = PDF
7 \}
```

3.7.2. Accessibility Support

The last example also demonstrates the `accsupp` key. The idea is to have something different visible in the PDF file compared to what you get when you select and copy the corresponding string. In the example visible string is a lowercase `pdf` in small caps while the string copied is an uppercase `PDF`.

For this to work you need to use the `package option accsupp`, too, which will load the package accsupp if it isn’t loaded by the user already. Then the key `accsupp` will set the `ActualText` property of `\BeginAccSupp`. Please refer to accsupp’s documentation for details. To see its effect copy `PDF` and paste it into a text file. You should get uppercase letters instead of lowercase ones.

`accsupp = \{⟨text⟩\}`

Key for `\DeclareAcronym` to set the `ActualText` property of `\BeginAccSupp` (see accsupp’s documentation for details) to be used for an acronym. It only has an effect when the package option `accsupp` is used, too.

`accsupp = true|false`  
Default: `false`

Activate `ACRO`’s accessibility support.

3.7.3. Tooltips

The idea of a tooltip is to have some text shown when you hover with the mouse over the short form of an acronym. This is only available in some PDF viewers, though. One possibility for such tooltips is loading the pdfcomment package [Kle18] and using its `\pdftooltip` macro.

`tooltip = true|false`  
Default: `false`

This option loads the pdfcomment package and sets the command for creating tooltips to `\pdftooltip`.

`tooltip-cmd = \{⟨control sequence⟩\}`  
Default: `\@firstoftwo`

This allows users using another macro for tooltips – maybe one provided by another package or some own macro. It needs to be a macro with two mandatory arguments, the first being the string typeset in the PDF, the second being the tooltip description text.

For using this with acronyms they have a property `tooltip` which can be used inside `\DeclareAcronym` for specifying the description text of the tooltip. If the `tooltip` package option is used but the property is not set for an acronym then the `long` property is used instead.

If an acronym is used inside of another acronym then the tooltips of the “inner” acronyms are disabled.
Adding Acronyms to the Index

\acro has the package option index. If it is used an index entry will be recorded every time an unstarred acronym command is used. The index entry will be \(\langle id\rangle@\langle short\rangle\), \(\langle sort\rangle@\langle short\rangle\) if the sort property has been set, \(\langle index-sort\rangle@\langle short\rangle\) if the property index-sort has been set, or \(\langle index\rangle\) if the property index has been set for the specific acronym. The short versions appearing there are formatted according to the chosen format of the corresponding acronym, of course.

This document demonstrates the feature. You can find every acronym that has been declared in the index. In order to allow flexibility the indexing command can be chosen both globally via package option and individually for every acronym. This would allow to add acronyms to a specific index if more than one index is used, for example with help of the imakeidx package.

I’m not yet convinced this is a feature many people if anyone needs and if they do if it is flexible enough. If you have any thoughts on this I’d appreciate an email.

Printing the List

Printing the whole list of acronyms is easy: just place \texttt{\printacronyms} where ever you want the list to be.

\texttt{\printacronyms[(options)]}

Print the list of acronyms.

The commands takes a few options, namely the following ones:

\texttt{\textbf{include-classes} = \{\langle list of classes\rangle\}}

Takes a comma-separated list of the classes of acronyms that should be in the list.

\texttt{\textbf{exclude-classes} = \{\langle list of classes\rangle\}}

Takes a comma-separated list of the classes of acronyms that should not be in the list. \textit{Note that this list overwrites any entries in include-classes!} If a class is both included and excluded then the corresponding acronyms will not be added to the list.

\texttt{\textbf{name} = \{\langle name of the list\rangle\}}

sets the name for the list.

\texttt{\textbf{heading} = \{\langle sectioning command without leading backslash\rangle\}} \hspace{1cm} \text{Default: section*}

Sets the sectioning command for the heading of the list. A special value is none which suppresses the heading.

\texttt{\textbf{sort} = true|false} \hspace{1cm} \text{Default: true}

Set sorting for this list only.

\texttt{\textbf{local-to-barriers} = true|false} \hspace{1cm} \text{Default: false}

This option can be used to create a list of only the acronyms of the current “barrier group”, see section 8.
You can see that the default layout is a description list with a \section* title. Both can be changed, see section 5.

The command \printacronyms needs two \LaTeX runs. This is a precaution to avoid error messages with a possibly empty list. But since almost all documents need at least two runs and often are compiled much more often than that, this fact shouldn’t cause too much inconvenience.

5. Options and Customization

5.1. General Options

There are a few options which change the general behaviour of ACRO. Underlined values are used if no value is given. ACRO’s package options are best set onyl via the setup command:
\texttt{\textbackslash{acsetup\{\langle options\rangle}}}

Set up \texttt{ACRO}.

\textbf{They might} work as package options to \texttt{\textbackslash{usepackage}} as well but \textit{I do not recommend it} and will likely remove this in a future version 3 of \texttt{ACRO}.

\texttt{messages = \{silent\}|\{loud\}} \hspace{1cm} \text{Default: loud}

Setting \texttt{messages = \{silent\}} will turn all of \texttt{ACRO}'s error messages into warnings and all of \texttt{ACRO}'s warnings into info messages. Be sure to check the log file carefully if you decide to set this option.

\texttt{single = \{true\}|\{false\}} \hspace{1cm} \text{Default: false}

If set to \texttt{true} an acronym that's used only once (with \texttt{\textbackslash{ac}}) in a document will only print the acronym in a specified form and will not be printed in the list.

\texttt{case-sensitive = \{true\}|\{false\}} \hspace{1cm} \text{Default: true}

If set to \texttt{true} the \texttt{\langle id\rangle} as used in \texttt{\DeclareAcronym} or \texttt{\textbackslash{ac}} is read \textit{case sensitive}, \textit{i.e.}, \texttt{id} would differ from \texttt{ID}. This is \texttt{ACRO}'s default behaviour.

If set to \texttt{true} \texttt{id} would be the same as \texttt{ID}. This can be useful when acronyms are used in page headings for example.

\texttt{single-form = \{long\}|\{short\}|\{alt\}|\{first\}} \hspace{1cm} \text{Default: long}

Determines how a single appearance of an acronym is printed if \texttt{single = \{true\}} has been chosen.

\texttt{hyperref = \{true\}|\{false\}} \hspace{1cm} \text{Default: false}

If set to \texttt{true} the short forms of the acronyms will be linked to their list entry.

\texttt{label = \{true\}|\{false\}} \hspace{1cm} \text{Default: false}

If set to \texttt{true} this option will place \texttt{\textbackslash{label}\{\langle prefix\rangle\langle id\rangle\}} the first time the acronym with \texttt{\textbackslash{ID} \langle id\rangle} is used.

\texttt{label-prefix = \{\langle text\}\}} \hspace{1cm} \text{Default: ac:}

The prefix for the \texttt{\textbackslash{label}} that is placed when option \texttt{label = \{true\}} is used.

\texttt{only-used = \{true\}|\{false\}} \hspace{1cm} \text{Default: true}

This option is \texttt{true} as default. It means that only acronyms that are actually used in the document are printed in the list. If \texttt{false}, all acronyms defined with \texttt{\DeclareAcronym} will be written to the list.

\texttt{mark-as-used = \{first\}|\{any\}} \hspace{1cm} \text{Default: any}

This option determines wether an acronym is mark as used when the \texttt{first} form is used the first time (with \texttt{ac}, \texttt{acf} or \texttt{acliffe} and their uppercase, plural and indefinite forms) or when any of the \texttt{ac-like} commands is used.
5. Options and Customization

**macros = true|false**  
Default: false  
If set to true this option will create a macro \(\text{id}\) for each acronym as a shortcut for \(\text{ac}\{\langle\text{id}\rangle\}\). Already existing macros will not be overwritten.

**xspace = true|false**  
Default: false  
If set to true this option will append \(\text{xspace}\) from the xspace package to the commands created with the macros option.

**strict = true|false**  
Default: false  
If set to true and the option macros = \{true\} is in effect then already existing macros will be overwritten.

**sort = true|false**  
Default: true  
If set to true the acronym list will be sorted alphabetically. The entries are sorted by their ID ignoring upper and lower case. This option needs the experimental package \text{l/three.taboldstylesort} (from the \text{l/three.taboldstyleexperimental} bundle) and can only be set in the preamble. If set to false the acronyms are listed in order of their declaration.

**display-foreign = true|false**  
Default: true  
When set to false any foreign property is ignored and foreign forms are not displayed.

**cite = all|first|none**  
Default: first  
This option decides whether citations that are added via cite are added to each first, every or no appearance of an acronym. If first is chosen, the option single = \{true\} is active and an acronym appears only once it still will get the citation.

**cite-cmd = \{\langle control sequence\rangle\}**  
Default: \cite  
This option determines which command is used for the citation. Each citation command that takes the cite key as argument is valid, for example biblatex’s \text{\footcite}.

**cite-connect = \{\langle code\rangle\}**  
Default: \nobreakspace  
Depending on the citation command in use a space should be inserted before the citation or maybe not (e.g. \text{\footcite}...). This option allows you to set this. Actually it can be used to place arbitrary code right before the citation.

**group-citation = true|false**  
Default: false  
Introduced in version 2.0  
If set to true the short form (or the long form) and the citation of an acronym will be printed together in parentheses when an acronym is used the first time.

**group-cite-cmd = \{\langle control sequence\rangle\}**  
Default: \cite  
Introduced in version 2.0  
This option determines which command is used for the citation when an acronym is used the first time and group-citation = \{true\}. Each citation command that takes the cite key as argument is valid, for example biblatex’s \text{\footcite}.

**group-cite-connect = \{\langle code\rangle\}**  
Default: \nobreakspace  
Introduced in version 2.0  
When group-citation = \{true\} this is inserted between the long/short form of the acronym and the citation within the parentheses.
5. Options and Customization

index = true|false  Default: false
If set to true an index entry will be recorded every time an unstarred acronym command is used for the corresponding acronym.

index-cmd = \{⟨control sequence⟩\}  Default: \index
Chooses the index command that is used when option index has been set to true.

accsupp = true|false  Default: false
Activates the access support as provided by the accsupp package.

tooltip = true|false  Default: false
Activates tooltip support for ACRO using the pdfcomment package.

tooltip-cmd = \{⟨control sequence⟩\}  Default: \@firstoftwo
A macro taking two mandatory arguments, the first being the short form of the acronym and the second being some tooltip description.

uc-cmd = \{⟨control sequence⟩\}  Default: \acfirstupper
The command that is used to capitalize the first word in the \Ac and the like commands. You can change it to another one like for example \makefirstuc\textsuperscript{10} or \MakeTextUppercase\textsuperscript{11}.

uppercase-short = true|false  Default: true
If set to true commands like \Ac or \Acp will uppercase the first letter of the short version.

\% with \acsetup{macros} activated:
we could have used these before: nato, ny

we could have used these before: NATO, NY

5.2. Options Regarding Acronyms

The options described in this section all influence the layout of one of the possible output forms of the acronyms.

short-format = \{⟨format⟩\}  (initially empty)
Sets a format for all short forms. For example short-format = \{\scshape\} would print all short forms in small caps.

long-format = \{⟨format⟩\}  (initially empty)
The same for the long forms.

foreign-format = \{⟨format⟩\}  (initially empty)
The format for the foreign entry when it appears as part of the first appearance of an acronym.

\textsuperscript{10} from the mfirstuc package
\textsuperscript{11} from the textcase package
5. Options and Customization

\texttt{single-format = \{format\}} 
\hspace{1cm} (initially empty)

The format for the acronym when it is used only once. If not specified the formatting according to \texttt{single-form} is used.

\texttt{first-long-format = \{format\}} 
\hspace{1cm} (initially empty)

The format for the long form on first usage (with \texttt{\ac}, \texttt{\acf} or \texttt{\acfl} and their uppercase, plural and indefinite forms).

\texttt{list-short-format = \{format\}} 
\hspace{1cm} (initially empty)

An extra format for the short entries in the list. If not used this is the same as \texttt{short-format}. Please be aware that a call of \texttt{short-format} after this one will overwrite it again.

\texttt{list-short-width = \{dim\}} 
\hspace{1cm} Default: 3em

This option controls the width reserved for the short forms of the acronyms in the \texttt{lof} list style.

\texttt{list-long-format = \{format\}} 
\hspace{1cm} (initially empty)

An extra format for the long entries in the list. If not used this is the same as \texttt{long-format}. Please be aware that a call of \texttt{long-format} after this one will overwrite it again.

\texttt{list-foreign-format = \{format\}} 
\hspace{1cm} Default: \texttt{\acroenparen}

The format for the foreign entry as it appears in the list. This may be code that ends with a macro that takes a mandatory argument.

\texttt{extra-format = \{format\}} 
\hspace{1cm} (initially empty)

The same for the extra information.

\texttt{first-style = default | empty | square | short | long | reversed | footnote | sidenote | footnote-reversed | sidenote-reversed} 
\hspace{1cm} Default: default

The basic style of the first appearance of an acronym. The value sidenote needs the command \texttt{\sidenote} to be defined (for example by the sidenotes package).

\texttt{extra-style = default | plain | comma | paren | bracket} 
\hspace{1cm} Default: default

Defines the way the extra information is printed in the list.

\texttt{plural-ending = \{short\}/\{long\}} 
\hspace{1cm} Default: s/s

With this option the default plural ending can be set. The appended \texttt{\{long\}} ending is optional. If you leave it (\textit{and} the /) the \texttt{\{short\}} ending is used for both short and long versions.
5. Options and Customization

5.3. Options Regarding the List

\begin{itemize}
\item \texttt{page-style = none|plain|comma|paren} \hspace{2cm} Default: none
\end{itemize}

If this option is set to a value other than none the page numbers of the an acronym appeared on are printed in the list. Please note that this is an experimental feature and might fail in quite a number of cases. If you notice anything please send me an email!

\begin{itemize}
\item \texttt{pages = all|first} \hspace{2cm} Default: all
\end{itemize}

If the option \texttt{page-style} has any value other than none this option determines wether all usages of the acronyms are listed or only the first time. Implicitly sets \texttt{label = \{true\}}.

\begin{itemize}
\item \texttt{page-name = \{(page name)\}} \hspace{2cm} Default: p. \textbackslash@, \textbackslash,
\end{itemize}

The “name” of the page label. This is automatically translated to the active language. However for the time being there are many translations missing, yet. Please notify me if you find your language missing.

\begin{itemize}
\item \texttt{pages-name = \{(page name plural)\}} \hspace{2cm} Default: pp. \textbackslash@, \textbackslash,
\end{itemize}

The “name” of the page label when there are more than one page. This is automatically translated to the active language. However for the time being there are many translations missing, yet. Please notify me if you find your language missing.

\begin{itemize}
\item \texttt{following-page = true|false} \hspace{2cm} Default: false
\end{itemize}

If set to true a page range in the list of acronyms that consists of two pages will be written by the first page and an appended f. This depends on the option \texttt{next-page}.

\begin{itemize}
\item \texttt{following-pages = true|false} \hspace{2cm} Default: false
\end{itemize}

If set to true a page range in the list of acronyms that set consists of more than two pages will be written by the first page and an appended ff. This depends on the option \texttt{next-pages}.

\begin{itemize}
\item \texttt{\textbackslash acsetup\{first-style=empty\}} \hspace{2cm} empty: NY
\item \texttt{\textbackslash acsetup\{first-style=footnote\}} \hspace{2cm} footnote: NY\textsuperscript{a}
\item \texttt{\textbackslash acsetup\{first-style=square\}} \hspace{2cm} square: New York [NY] [Wik12]
\item \texttt{\textbackslash acsetup\{first-style=short\}} \hspace{2cm} short: NY [Wik12]
\item \texttt{\textbackslash acsetup\{first-style=long\}} \hspace{2cm} long: New York [Wik12]
\item \texttt{\textbackslash acsetup\{first-style=reversed\}} \hspace{2cm} reversed: NY (New York) [Wik12]
\item \texttt{\textbackslash acsetup\{first-style=footnote-reversed\}} \hspace{2cm} footnote-reversed: New York\textsuperscript{b}
\end{itemize}

\textsuperscript{a} New York [Wik12] \\
\textsuperscript{b} NY [Wik12]
6. Trailing Tokens and Special Action

following-pages\* = true|false \hspace{1cm} Default: false
If set to true this sets both options following-page = {true} and following-pages = {true}. false sets following-page = {false} and following-pages = {false}.

next-page = \{\langle text\rangle\} \hspace{1cm} Default: \,f.\@
Appended to a page number when following-page is set to true and the range is only 2 pages long. This is automatically translated to the active language. However, for the time being there are many translations missing, yet. Please notify me if you find your language missing.

next-pages = \{\langle text\rangle\} \hspace{1cm} Default: \,ff.\@
Appended to a page number when following-pages is set to true and the range is more than 2 pages long. This is automatically translated to the active language. However, for the time being there are many translations missing, yet. Please notify me if you find your language missing.

list-style = description | lof | longtable | extra-longtable | extra-longtable-rev | extra-tabular|extra-tabular-rev|tabular|toc \hspace{1cm} Default: description
Choose with which style the list of acronyms should be typeset. If you choose (longtable), extra-longtable or extra-longtable-rev you have to load the longtable [Car/one.taboldstyle/nine.taboldstyle] package in your preamble. The values extra-\langle something\rangle put the extra information in a column of it own. Be aware that per default all extra-table styles only use 1 columns. Since acronym descriptions can easily get longer that a line you should probably define your own style if you want to use them. See section 13.2 on page 35 for details.

list-heading = chapter | chapter* | section | section* | subsection | subsection* | subsubsection|subsubsection*|addchap|addsec|none \hspace{1cm} Default: section*
The heading type of the list. The last two only work with a KOMA-Script class that also defines the appropriate command. A special value is none which suppresses the heading.

list-name = \{\langle list name\rangle\} \hspace{1cm} Default: Acronyms
The name of the list. This is what’s written in the list-heading. This is automatically translated to the active language. However, for the time being there are many translations missing, yet. Please notify me if you find your language missing.

list-caps = true|false \hspace{1cm} Default: false
Print the first letters of the long form capitalized.

6. Trailing Tokens and Special Action

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

A token is made known to \acro with the following macro:

\acroRegisterTrailing\{token\}\{\langle name\rangle\}
This registers the token \langle token\rangle so \acro looks if it follows directly after an acronym macro. \langle name\rangle is the internal name for this token.
The \acro package already registers the above mentioned tokens:

\begin{itemize}
  \item \acroRegisterTrailing {dot}
  \item \acroRegisterTrailing {dash}
  \item \acroRegisterTrailing \parens{babel-hyphen}
\end{itemize}

If a token is registered it doesn’t mean that \acro looks for it. The token must first be activated for this:

activate-trailing-tokens = \{\csv list of token names\}

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.

deactivate-trailing-tokens = \{\csv list of token names\}

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

All of the above on its own does nothing visible. However: inside of an acronym, \ie for example inside the long or the short form it can be tested for those trailing tokens:

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

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

\acro uses this to define to further macros:

\acdot

Inserts a . if no dot follows.

\acspace

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

The definitions are equivalent\footnote{Not quite. \acro’s definitions are engine protected.} to the following code:

\begin{itemize}
  \item \newcommand*{\acdot}{\aciftrailing{dot}\{}{.\@}}
  \item \newcommand*{\acspace}{\aciftrailing{dash,babel-hyphen}\{}{\space}}
\end{itemize}

This could be used to define an acronym as follows:

\begin{itemize}
  \item \DeclareAcronym{etc}{
    \short = \{\textit{etc}\acdot\},
    \long = \{\textit{et cetera}\},
    \short-plural = , \long-plural =
\end{itemize}
If now you somewhere use \texttt{\textbackslash ac(etc).} there won’t be two dots printed. The command \texttt{\textbackslash acspace} is used already in the definition of the first appearance of a macro. Let’s say you’re a German chemist and you have

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

and you use it the first time like this:

\begin{verbatim}
\ac{PU}-Hartschaum
\end{verbatim}

then according to German orthography and typesetting rules this should be printed as “Polyurethan(\text{PU})-Hartschaum” \textit{i.e.}, with \textit{no} space between long and short form. This is exactly what happens if you say

\begin{verbatim}
\acsetup{activate-trailing-tokens = \{\text{dash,babel-hyphen}\}}
\end{verbatim}

in the preamble.

### 7. About Page Ranges

If you enable the \texttt{page-style} option \texttt{ACRO} adds page numbers to the list of acronyms. In version 0.1.1 it would add a page reference for an acronym in the list of acronyms that used \texttt{\textbackslash pagereref} to refer to the first appearance of an acronym. This is retained using \texttt{pages = \{first\}}. Version 1.0 uses a different approach that doesn’t use a label but instead will list all pages an acronym appeared on. With hyperref the pages are referenced using \texttt{\hyperpage}. There are some options that control how this list will be typeset, e.g., \texttt{following-page}, \texttt{next-pages} or the option \texttt{page-style} itself. It is important to mention that the page list will always take at least two compilation runs until changes in the options or the actual page
8. Dividing Your Document Into Pieces – Creating Local Lists

\acro introduces the concept of \textit{barriers} which can divide the document into parts. It is possible to create lists of only those acronyms used between two such barriers.

\texttt{\textbackslash acbarrier}

Sets a barrier at the point of use in the document. The begin and the end of the document mark implicit barriers.

\begin{itemize}
  \item \texttt{use-barriers = \texttt{true}|false} \hspace{1cm} \texttt{Default: false}
  \item \texttt{reset-at-barriers = \texttt{true}|false} \hspace{1cm} \texttt{Default: false}
  \item \texttt{local-to-barriers = \texttt{true}|false} \hspace{1cm} \texttt{Default: false}
\end{itemize}

If you want to use barriers and local lists you have to activate the feature first. This should be set in the preamble in order to work reliably. Make sure to watch out for log file messages asking you to rerun.

If this option is set to \texttt{true} \texttt{\acbarrier} implicitly calls \texttt{\acresetall}.

This option can \textit{only} be used as option to the \texttt{\printacronyms} command. It then prints a list of only the acronyms of the current "barrier group".

\begin{itemize}
  \item \texttt{\acbarrier}
  \item \texttt{\printacronyms[local-to-barriers]}
  \item \texttt{\ac{ctan}} and \texttt{\ac{lppl}}
  \item \texttt{\acbarrier}
\end{itemize}

9. Language Support

\acro detects if packages babel [Bra19] or polyglossia [Bra19] are being loaded and tries to adapt certain strings to match the chosen language. However, due to my limited language knowledge only a few translations are provided. I’ll show how the English translations are defined so you can add the translations to your preamble if needed. Even better would be you’d send me a short email to contact@mychemistry.eu with the appropriate translations for your language and I’ll add them to \acro.
10. hyperref Support

The option \texttt{hyperref = {true}} adds internal links from all short (or alternative) forms to their respective list entries. Of course this only works if you have loaded the \texttt{hyperref} package in your preamble. You should use this option with care: if you don’t use \texttt{\textbackslash printacronyms} anywhere this option will result in loads of \texttt{hyperref} warnings. Also printing several lists can result in warnings if don’t clearly separate the lists into different classes. If an acronym appears in more than one list there will also be more than one hypertarget for this acronym.

Using \texttt{hyperref} will also add \texttt{\textbackslash hyperpage} to the page numbers in the list (provided they are displayed in the style chosen). Like with an index the references will thus not point to the acronyms directly but to the page they’re on.

11. Defining Own Acronym Macros

The commands explained in section 2.3 on page 7 have all been defined with a dedicated command – there is a family of dedicated commands, actually:

\begin{quote}
\texttt{\textbackslash NewAcroCommand{\langle cs\rangle}{\langle code\rangle}}
\end{quote}

Defines a new \texttt{acro} acronym command \texttt{(cs)}. This sets up the necessary framework needed by acronym commands and defines \texttt{(cs)} with an optional star argument and a mandatory argument for the acronym id using \texttt{xparse’s} \texttt{\textbackslash NewDocumentCommand}. Inside \texttt{(code)} one can refer to the \texttt{ID} \texttt{(id)} with \texttt{#1}.

\begin{quote}
\texttt{\textbackslash RenewAcroCommand{\langle cs\rangle}{\langle code\rangle}}
\end{quote}

Defines a new \texttt{acro} acronym command \texttt{(cs)}. This sets up the necessary framework needed by acronym commands and defines \texttt{(cs)} with an optional star argument and a mandatory argument for the acronym id using \texttt{xparse’s} \texttt{\textbackslash RenewDocumentCommand}. Inside \texttt{(code)} one can refer to the \texttt{ID} \texttt{(id)} with \texttt{#1}.

\begin{quote}
\texttt{\textbackslash DeclareAcroCommand{\langle cs\rangle}{\langle code\rangle}}
\end{quote}

Defines a new \texttt{acro} acronym command \texttt{(cs)}. This sets up the necessary framework needed by acronym commands and defines \texttt{(cs)} with an optional star argument and a mandatory argument for the acronym id using \texttt{xparse’s} \texttt{\textbackslash DeclareDocumentCommand}. Inside \texttt{(code)} one can refer to the \texttt{ID} \texttt{(id)} with \texttt{#1}.

\begin{quote}
\texttt{\textbackslash ProvideAcroCommand{\langle cs\rangle}{\langle code\rangle}}
\end{quote}

Defines a new \texttt{acro} acronym command \texttt{(cs)}. This sets up the necessary framework needed by
Defining Own Acronym Macros

acronym commands and defines \cs with an optional star argument and a mandatory argument for the acronym id using xparse’s \ProvideDocumentCommand. Inside \code one can refer to the id \id with #1.

Inside these macros one can use a number of low-level expl3 commands.\footnote{Which is why you need to use them inside an expl3 programming environment. This means in the preamble surround the definitions with \ExplSyntaxOn and \ExplSyntaxOff.}

**Acronym fetching commands**

\acro_use:n {\id}
Fetched the acronym using either the first or the short form depending on earlier uses.

\acro_short:n {\id}
Fetched the short form of the acronym.

\acro_long:n {\id}
Fetched the long form of the acronym.

\acro_alt:n {\id}
Fetched the alternative short form of the acronym.

\acro_foreign:n {\id}
Fetched the foreign property of the acronym if available.

\acro_extra:n {\id}
Fetched the extra property of the acronym if available.

**Acronym setup commands**

\acro_first_upper:
\acro setup command which tells the macros above that we want to uppercase the first letter of the long version. Should be used before one of the acronym fetching commands.

\acro_plural:
\acro setup command which tells the macros above that we want to use plural forms. Should be used before one of the acronym fetching commands.

\acro_indefinite:
\acro setup command which tells the macros above that we want to add the indefinite article. Should be used before one of the acronym fetching commands.

\acro_cite:
\acro setup command which tells the macros above that we want to add the citation in any case independent of the option cite. Should be used before one of the acronym fetching commands.
11. Defining Own Acronym Macros

\acro_no_cite:
ACRO setup command which tells the macros above that we want to have no citation independent of the option cite. Should be used before one of the acronym fetching commands.

\acro_index:
ACRO setup command which tells the macros above that we want to add an index entry in any case independent of the option index. Should be used before one of the acronym fetching commands.

\acro_reset_specials:
This macro is called implicitly by \NewAcroCommand and \NewPseudoAcroCommand. If you plan to define an ACRO command by yourself using \NewDocumentCommand this should be the first macro after \acro_begin:. It ensures that in nested acronyms the inner acronyms don’t inherit indefinite articles, uppercasing, endings...

Additional macros for further uses

\acro_begin:
When an acronym macro is defined "by hand", i.e., not using \NewAcroCommand then this must be the first macro in the code. Must have a matching \acro_end:.

\acro_end:
When an acronym macro is defined "by hand", i.e., not using \NewAcroCommand then this must be the last macro in the code. Must have a matching \acro_begin:.

\acro_check_and_mark_if:nn {⟨boolean expression⟩} {⟨id⟩}
Checks if the acronym with the ID ⟨id⟩ exists and marks it as used when ⟨boolean⟩ expression evaluates to true. This macro is used inside \NewAcroCommand and friends implicitly.

\acro_check_acronym:nn {⟨id⟩} {true|false}
Checks if the acronym with the ID ⟨id⟩ exists and marks it as used if true or doesn’t. This macro is used inside \acro_check_and_mark_if:nn.

\acro_use_acronym:n {true|false}
Tell \acro_use:n and similar commands wether to mark the acronym as used or not. This macro is used inside \acro_check_acronym:nn. If this macro is used explicitly it should be used before \acro_use:n (or a similar command) otherwise it has no effect. An acronym marked as used cannot be unmarked.

\acro_mark_as_used:n {⟨id⟩}
Explicitly use the acronym with the ID ⟨id⟩. This is the expl3 macro applied to all entries in \acuse.

*\acro_if_acronym_used:nTF {⟨id⟩} {⟨true⟩} {⟨false⟩}
The code-level version of \acifused. This macro is expandable.
11. Defining Own Acronym Macros

\acro_for_all_acronyms_do:n \{⟨code⟩\}
Loops over all acronyms known when the macro is used. Inside of ⟨code⟩ you can refer to the 1D ⟨id⟩ of an acronym with #1.

\acro_barrier:
The code-level version of \acbarrier.

\acro_switch_off:
The expl3 version of \acswitchoff.

\acro_switch_on:
The expl3 version of \acswitchon.

\acro_add_action:n \{⟨code⟩\}
Adds code to \acro_get:n. Inside of ⟨code⟩ you can refer to the 1D of the acronym with #1. The \acro_get:n command is called the first macro that is called by \acro_use:n which for example enables conditionals to test for certain acronym properties.

\acro_property_get:nnTF \{⟨id⟩\} \{⟨property⟩\} \{⟨true⟩\} \{⟨false⟩\}
Fetches the property ⟨property⟩ of the acronym ⟨id⟩ and stores it in a tokenlist variable \l__acro_⟨property⟩_tl where all dashes in the property names are replaced with underscores. ⟨true⟩ is placed in the input stream if the property had been set, ⟨false⟩ otherwise.

\acro_property_get:nn \{⟨id⟩\} \{⟨property⟩\}
Like \acro_property_get:nnTF, but without the ⟨true⟩ and ⟨false⟩ arguments.

\acro_property_if_set:nnTF \{⟨id⟩\} \{⟨property⟩\} \{⟨true⟩\} \{⟨false⟩\}
Checks if the property ⟨property⟩ of the acronym ⟨id⟩ is set and places ⟨true⟩ in the input stream if yes and ⟨false⟩ otherwise.

Examples

The usage of above macros is best explained with a few examples. The following definition is done by \acro:

\NewAcroCommand \ac \{ \acro_use:n \{#1\} \}

An equivalent definition for \ac would be

\NewDocumentCommand \ac \{\sm\}
{\acro_begin:
  \acro_reset_specials:
  \acro_check_and_mark_if:nn \{#1\} \{#2\}
  \acro_use:n \{#2\}
\acro_end:
}
About Plural Forms, Possessive Forms and Similar Constructs – the Concept of Endings

which should explain what the actual framework is which \NewAcroCommand adds. Other definitions by \acro are for example the following ones:

\acro,
\acro
\acro
\acro
\acro,
\acro
\acro
\acro,
\acro
\acro
\acro
\acro

12. About Plural Forms, Possessive Forms and Similar Constructs – the Concept of Endings

\acro has a concept of endings. All of \acro’s plural options are defined by saying

\ProvideAcroEnding {plural} {s} {s}

The command’s syntax and what it does is as follows:

\ProvideAcroEnding{(name)}{(short default)}{(long default)}

This macro defines the options

- \{(name)\}-ending,
- short-\{(name)\}-ending,
• \textit{alt-(name)-ending},
• \textit{long-(name)-ending} and
• \textit{foreign-(name)-ending}.

It also defines the acronym properties

• \textit{short-(name)},
• \textit{short-(name)-form},
• \textit{alt-(name)},
• \textit{alt-(name)-form},
• \textit{long-(name)},
• \textit{long-(name)-form},
• \textit{foreign-(name)} and
• \textit{foreign-(name)-form}.

Additionally it defines a setup macro as described in section 11 on page 29, \texttt{\acro_\langle name\rangle}: If \langle name\rangle contains a - (dash) it is replaced by _ before \texttt{\acro_\langle name\rangle} is built. So if you choose \texttt{my-name} the corresponding macro is named \texttt{\acro_my\_name}: If you use any other non-letters you are on your own. If you use the command with the same \langle name\rangle a second time the command only resets the defaults.

Note that you \textbf{must use} \texttt{\ProvideAcroEnding} before any acronym definition!

This could be used together with the macros described in section 11 on page 29 for adding support for possessive forms:

\begin{verbatim}
\ExplSyntaxOn
\% this now only works because I've used the same already in the preamble so
\% it does nothing here:
\ProvideAcroEnding \ac{possessive} {'s} {'s}
\ProvideAcroCommand \acg
\{ \acro_possessive:
\acro_use:n \{#1} \}
\ExplSyntaxOff
\end{verbatim}

The \texttt{\acg(cd)} booklet says...

Please note that different endings are cumulative which you probably want to avoid! Imagine a macro
13. More on Customization

13.1. Background Information

Several of \acro\'s objects are customized using templates. For each of these objects it is possible
to define own templates.\footnote{This requires some knowledge of xtemplate and \texttt{expl3}. Plans are to provide a documented interface for users of \acro in the future.} Possibly more interesting: it is easily possible to define further instances of an object using a certain template. How this works is explained in the following sections. However, the basics are always the same. There is a command

```latex
\DeclareAcro\langle object type\rangle\Style\{\langle name\rangle\}\{\langle template\rangle\}\{\langle options\rangle\}
```

which allows to define a new style (i.e., instance) for the object \langle object type\rangle using the template \{\langle template\rangle\}.

13.2. Lists

13.2.1. Own List Style

The different existing list styles are all built from four different templates, \texttt{list}, \texttt{list-of}, \texttt{table} and \texttt{extra-table}. Those templates are defined with the help of the \texttt{xtemplate} package (from [L3P]). Each of these templates has a few options which are described in table 1 on the following page. New list styles now are defined via the following macro:

```latex
\DeclareAcroListStyle\{\langle name\rangle\}\{\langle template\rangle\}\{\langle options\rangle\}
```

Declares a new \acro list style \langle name\rangle. \langle name\rangle will be the value which can be chosen in the option \texttt{list-style}. \langle template\rangle is the name of the template to be used by the style. Available templates are listed in table 1. \langle options\rangle are the option settings for the corresponding template.

For defining new styles you need some information on what the different templates and options do:
### Table 1: Available List Templates and Their Options

<table>
<thead>
<tr>
<th>Template</th>
<th>Option</th>
<th>Option Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>list</td>
<td>list</td>
<td>tokenlist</td>
<td>description</td>
</tr>
<tr>
<td></td>
<td>foreign-sep</td>
<td>tokenlist</td>
<td>\space</td>
</tr>
<tr>
<td></td>
<td>reverse</td>
<td>boolean</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>before</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>after</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td>list-of</td>
<td>style</td>
<td>tokenlist</td>
<td>toc</td>
</tr>
<tr>
<td></td>
<td>foreign-sep</td>
<td>tokenlist</td>
<td>\space</td>
</tr>
<tr>
<td></td>
<td>reverse</td>
<td>boolean</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>before</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>after</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td>table</td>
<td>table</td>
<td>tokenlist</td>
<td>tabular</td>
</tr>
<tr>
<td></td>
<td>table-spec</td>
<td>tokenlist</td>
<td>lp{.7\linewidth}</td>
</tr>
<tr>
<td></td>
<td>foreign-sep</td>
<td>tokenlist</td>
<td>\space</td>
</tr>
<tr>
<td></td>
<td>reverse</td>
<td>boolean</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>before</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>after</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td>extra-table</td>
<td>table</td>
<td>tokenlist</td>
<td>tabular</td>
</tr>
<tr>
<td></td>
<td>table-spec</td>
<td>tokenlist</td>
<td>llll</td>
</tr>
<tr>
<td></td>
<td>foreign-sep</td>
<td>tokenlist</td>
<td>\space</td>
</tr>
<tr>
<td></td>
<td>reverse</td>
<td>boolean</td>
<td>false</td>
</tr>
<tr>
<td></td>
<td>before</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>after</td>
<td>tokenlist</td>
<td></td>
</tr>
</tbody>
</table>
13. More on Customization

- The option \texttt{list} of the \texttt{list} template sets the list environment. This must be a classic \LaTeX list where items are listed with \texttt{\item}. In those lists short entries will always be fed as optional argument to \texttt{\item}:
  \texttt{\item[(short)] (long) (extra) (page)}

- The template \texttt{list-of} simulates a table of contents or a list of figures. This can be chosen by setting the option \texttt{style} to either \texttt{toc} or \texttt{lof}.

- The template \texttt{table} typesets the list in a table with two columns:
  \texttt{(short) & (long) (extra) (page) \tabularnewline}

- The template \texttt{extra-table} typesets the list in a table with four columns:
  \texttt{(short) & (long) & (extra) & (page) \tabularnewline}

- The option \texttt{foreign-sep} is the code inserted between long form and foreign entry (if a foreign entry is present).

- The options \texttt{before} and \texttt{after} are inserted directly before and after the complete list.

- The option \texttt{reverse} switches the place of \texttt{(long)} with \texttt{(extra)}.

- The option \texttt{table-spec} sets the column types for the table templates. It must correspond to the number of columns the corresponding template uses.

As an example let’s define a style \texttt{longtabu} which uses the corresponding table environment from the package tabu [Che19]:

\begin{verbatim}
\usepackage{tabu,longtable}
\DeclareAcroListStyle{longtabu}{table}{
table = longtabu, 
table-spec = @{}>{\bfseries}lX@{}
}
\acsetup{list-style=longtabu}
\end{verbatim}

As another example let’s define a new list with the help of the \texttt{enumitem} package [Bez19]:

\begin{verbatim}
% preamble:
% \usepackage{enumitem}
\newlist{acronyms}{description}{1}
\newcommand*{\addcolon}[1]{#1:}
\setlist[acronyms]{
  labelwidth = 3em, 
  leftmargin = 3.5em, 
  noitemsep, 
  itemindent = 0pt, 
  font=\addcolon}
\DeclareAcroListStyle{mystyle}{list}{ list = acronyms }
\end{verbatim}

37
13. More on Customization

\acsetup{ list-style = mystyle }

This would look as follows:

Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>Compact Disc</td>
</tr>
<tr>
<td>ECU</td>
<td>Steuergerät (Electronic Control Unit)</td>
</tr>
<tr>
<td>ID</td>
<td>identification string</td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographic Experts Group</td>
</tr>
<tr>
<td>LA</td>
<td>Los Angeles</td>
</tr>
<tr>
<td>MP</td>
<td>Member of Parliament</td>
</tr>
<tr>
<td>NATO</td>
<td>North Atlantic Treaty Organization. deutsch: Organisation des Nordatlantikvertrags</td>
</tr>
<tr>
<td>NY</td>
<td>New York</td>
</tr>
<tr>
<td>PDF</td>
<td>Portable Document Format</td>
</tr>
<tr>
<td>SW</td>
<td>Sammelwerk</td>
</tr>
<tr>
<td>ST</td>
<td>Some Test</td>
</tr>
<tr>
<td>UFO</td>
<td>unidentified flying object</td>
</tr>
</tbody>
</table>

13.2.2. Own List Heading Command

With the option list-heading you can choose which command prints the heading of the list. If you need a different choice than what’s already provided you can use the following command to define a new option:

\DeclareAcroListHeading{⟨name⟩}{⟨control sequence⟩}

Defines a new value ⟨name⟩ for the option list-heading. ⟨control sequence⟩ must be a control sequence which takes one mandatory argument.

As an example here is how the value section is defined:

\[ \text{\textbackslash DeclareAcroListHeading}{\text{section}}{\textbackslash section} \]

13.3. First Styles

The first styles define how an acronym is typeset when it is used for the first time. It is set with the option first-style. Legal values for this option are defined with the following command:

\DeclareAcroFirstStyle{⟨name⟩}{⟨template⟩}{⟨options⟩}

Declares a new ACRO first style ⟨name⟩. ⟨name⟩ will be the value which can be chosen in the option first-style. ⟨template⟩ is the name of the template to be used by the style. Available templates are listed in table 2. ⟨options⟩ are the option settings for the corresponding template.

Here are two examples of the already available styles and how they are defined:
13. More on Customization

\begin{verbatim}
\DeclareAcroFirstStyle{short}{inline}{
  only-short = true ,
  brackets   = false
}
\DeclareAcroFirstStyle{sidenote-reversed}{note}{
  note-command = \sidenote{#1} ,
  reversed     = true
}
\end{verbatim}

\begin{table}[h]
\centering
\begin{tabular}{lcl}
\hline
Template & Option & Option Type & Default \\
\hline
inline   & brackets     & boolean    & true \\
         & brackets-type & tokenlist  & ()  \\
         & only-short   & boolean    & false \\
         & only-long    & boolean    & false \\
         & reversed     & boolean    & false \\
         & between      & tokenlist  & \\
         & foreign-sep  & tokenlist  & ,~  \\
\hline
note     & use-note     & boolean    & true \\
         & note-command & function   & \footnote{#1} \\
         & reversed     & boolean    & false \\
         & foreign-sep  & tokenlist  & ,~  \\
\hline
\end{tabular}
\caption{Available First Style Templates and Their Options}
\end{table}

13.4. Extra Styles

The extra styles define how the extra information of an acronym is typeset in the list. It is set with the option \texttt{extra-style}. Legal values for this option are defined with the following command:

\begin{verbatim}
\DeclareAcroExtraStyle{\{name\}}{\{template\}}{\{options\}}
\end{verbatim}

Declares a new \texttt{ACRO} extra style \texttt{\{name\}}. \texttt{\{name\}} will be the value which can be chosen in the option \texttt{extra-style}. \texttt{\{template\}} is the name of the template to be used by the style. Available templates are listed in table 3. \texttt{\{options\}} are the option settings for the corresponding template.

Here are two examples of the already available styles and how they are defined:

\begin{verbatim}
\DeclareAcroExtraStyle{default}{inline}{
  brackets   = false ,
  punct      = true ,
  punct-symbol = .
}\end{verbatim}
### 13. More on Customization

**Available Extra Style Templates and Their Options**

<table>
<thead>
<tr>
<th>Template</th>
<th>Option</th>
<th>Option Type</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>inline</td>
<td>punct</td>
<td>boolean</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>punct-symbol</td>
<td>tokenlist</td>
<td></td>
</tr>
<tr>
<td></td>
<td>brackets</td>
<td>boolean</td>
<td>true</td>
</tr>
<tr>
<td></td>
<td>brackets-type</td>
<td>tokenlist</td>
<td>()</td>
</tr>
</tbody>
</table>

```latex
5 }\DeclareAcroExtraStyle{paren}{inline}{
7   brackets = true ,
8   punct = true ,
9   punct-symbol =
10 }\end{verbatim}
```

### 13.5. Page Number Styles

The page number styles define how the page numbers where acronyms have been used are typeset in the list. It is set with the option `page-style`. Legal values for this option are defined with the following command:

```latex
\DeclareAcroPageStyle{⟨name⟩}{⟨template⟩}{⟨options⟩}
```

Declares a new `acro` extra style `⟨name⟩`. `⟨name⟩` will be the value which can be chosen in the option `page-style`. `⟨template⟩` is the name of the template to be used by the style. Available templates are listed in table 4. `⟨options⟩` are the option settings for the corresponding template.

Here are two examples of the already available styles and how they are defined:

```latex
1 \DeclareAcroPageStyle{default}{inline}{
2   punct = true ,
3   punct-symbol = .
4 }\end{verbatim}
5 \DeclareAcroPageStyle{paren}{inline}{
6   brackets=true ,
7   punct-symbol =~
8 }\end{verbatim}
```

### 13.6. Configuration Files

If you repeatedly have the same setup and definitions for `acro` in your preamble you might want to place those in a configuration file. If `acro` finds a file named `acro.cfg` present it inputs

15. For example defining new endings, `acro` commands, list styles, …
**A. All Acronyms Used in this Documentation**

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CD</td>
<td>Compact Disc, pp. 8, 10, 34 f.</td>
<td></td>
</tr>
<tr>
<td>ECU</td>
<td>Steuergerät (Electronic Control Unit), pp. 11 f.</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>identification string, pp. 2, 4 f., 13, 15, 20 f., 29, 31 f.</td>
<td></td>
</tr>
<tr>
<td>JPEG</td>
<td>Joint Photographic Experts Group, p. 10</td>
<td></td>
</tr>
<tr>
<td>LA</td>
<td>Los Angeles, p. 14</td>
<td></td>
</tr>
<tr>
<td>MP</td>
<td>Member of Parliament, p. 10</td>
<td></td>
</tr>
<tr>
<td>NATO</td>
<td>North Atlantic Treaty Organization, <em>deutsch</em>: Organisation des Nordatlantikvertrags, pp. 11, 22</td>
<td></td>
</tr>
<tr>
<td>NY</td>
<td>New York, pp. 6, 10, 14 f., 22 f.</td>
<td></td>
</tr>
<tr>
<td>PDF</td>
<td>Portable Document Format, pp. 5, 16 f.</td>
<td></td>
</tr>
<tr>
<td>SW</td>
<td>Sammelwerk, p. 10</td>
<td></td>
</tr>
<tr>
<td>ST</td>
<td>Some Test, p. 5</td>
<td></td>
</tr>
<tr>
<td>UFO</td>
<td>unidentified flying object, p. 13</td>
<td></td>
</tr>
</tbody>
</table>

**B. References**

B. References

url: http://mirror.ctan.org/macros/latex/required/babel/.

url: http://mirror.ctan.org/macros/latex/required/tools/.

url: http://mirror.ctan.org/macros/latex/contrib/polyglossia/.

url: http://mirror.ctan.org/macros/latex/contrib/tabu/.

url: http://mirror.ctan.org/macros/latex/contrib/pdfcomment/.

url: http://mirror.ctan.org/macros/latex/contrib/l3packages/.

url: http://mirror.ctan.org/macros/latex/contrib/translations/.

url: http://mirror.ctan.org/macros/latex/contrib/glossaries/.

C. Index

A
\Ac ................................. 7, 13, 22, 33
\ac .................. 5, 7 ff., 11 ff., 15 ff., 20 ff., 23, 27 ff., 32
\Aca .................................. 8
\aca .................................. 7 ff., 11
\Acap .................................. 8
\acap .................................. 8
\acbarrier ................................ 28, 32
accsupp ............................. 17, 22
accsupp (property) ........... 5, 17
accsupp (package) .......... 5, 17, 22
\acfot .................................. 26
\Acf .................................. 8
\acf.................................. 5, 8 ff., 20, 23 ff.
\acfirstupper .......................... 13, 22
\acflike ................................. 5, 13 ff., 20, 23
\Acfp .................................. 8
\acfp .................................. 8
\acfplike ................................ 14
\acg .................................... 34
\acifirst ................................ 15
\aciftrailing .......................... 26
\acifused ............................ 15, 31
\Acl .................................. 7
\acl .................................. 7 ff., 15
\Aclp .................................. 8, 33
\aclp .................................. 8 ff.
\Acp .................................. 8, 13, 22, 33
\acp .................................. 8, 33
\acpdfstring ............................. 16
\acpdfstringplural .................... 16
\acpsingle ................................ 14
\acreset ................................ 15
\acresetall .......................... 15, 28
\acroenparen .......................... 12, 23
\AcroRegisterTrailing ........... 25 ff.
\Acs ................................. 7, 13
\acs .................. 7 ff., 11, 15 ff.
\acssetup ............. 14, 19 ff., 24, 27, 37 ff.
\Acsingle .................... 14
\acsingle .................... 14
\Acsp .................................. 8, 13
\acsp .................................. 8 ff., 16
\acspace ................................ 26 ff.
\acswitchoff ...................... 16, 32
\acswitchon ...................... 16, 32
activate-trailing-tokens .......... 26
acuse .................................. 16, 31
acuseall ............................. 16
after-citation (property) .......... 4
alt (property) .................... 4, 7, 10
alt-format (property) .......... 4
alt-indefinite (property) ........ 4, 12
alt-plural (property) .......... 3
alt-plural-form (property) ........ 3

B
babel ................................ 4, 12, 28
babel (package) ................ 4, 12, 28
before-citation (property) .......... 4
BEZOS, Javier ................. 4, 12, 28, 37
BRAAMS, Johannes ............. 4, 12, 28

C
Carlisle, David ................ 25
case-sensitive .................. 2, 20
CD .................................. 9 ff., 34 ff.
Charette, François ........ 4, 12
Chermet, Florent ........ 37
cite ................................ 21, 30 ff.
cite (property) ................ 4, 6, 21
cite-cmd .......................... 21
cite-connect ........................ 21
class (property) ................ 4, 14
cTAN .............................. 2
CTAN .................................. 28

decactivate-trailing-tokens .......... 26
\DeclareAcroCommand .......... 29
\DeclareAcroExtraStyle .......... 39 ff.
\DeclareAcroFirstStyle .......... 38 ff.
\DeclareAcroListHeading .......... 38
Index

\ DeclareAcroListStyle \ DeclareAcronym \ DeclareAcroPageStyle \ display-foreign

E
ECU \ enumitem \ enumitem (package) \ exclude-classes \ expl3 (package) \ extra (property) \ extra-format \ extra-style

F
first-long-format \ first-long-format (property) \ first-style (property) \ following-page \ following-pages \ foreign (property) \ foreign-format \ foreign-lang (property) \ foreign-plural (property) \ foreign-plural-form (property) \ format-include-endings

G
group-citation \ group-cite-cmd \ group-cite-connect

H
heading \ hyperref \ hyperref (package)

I\ \i\ac \ \iac \ \iacl

J
ID \ include-classes \ index \ index (property) \ index-cmd (property) \ index-sort (property)

K
JPEG \ Kleber, Josef

L\ l\experimental (package) \ l\keysze (package) \ l\packages \ l\sort (package)

\ LA \ label \ label-prefix \ list (property) \ list-caps \ list-foreign-format \ list-heading \ list-long-format \ list-name \ list-short-format \ list-short-width \ list-style \ local-to-barriers \ log \ long (property) \ long-format \ long-indefinite (property) \ long-plural (property) \ long-plural-form (property) \ long-post (property) \ long-pre (property) \ longtable \ longtable (package) \ LPPL
Index

M
macros ........................................ 21
\makefirstuc .............................. 13
mark-as-used ............................ 20
messages .................................. 20
mfirstuc (package) .................... 13, 22
MP ......................................... 10

N
name ......................................... 18
NATO ......................................... 11, 22
New York City .......................... 6, 14 f., 24
\NewAcroCommand ..................... 29, 31 ff., 35
\NewPseudoAcroCommand ............ 31
next-page ................................ 24 f.
next-pages ................................ 24 f., 27
Niederberger, Clemens ............. 2
NY ........................................ 6, 10, 14 f., 22, 24

O
only-used .................................. 20
OT ........................................ 6

P
page-name ................................ 24
page-style ................................ 24, 27, 40
pages ....................................... 24, 27
pages-name ................................ 24
PDF ........................................ 5, 16 f.
pdfcomment .............................. 17
pdfcomment (package) ................ 17, 22
pdfstring (property) ................ 5, 16
plural-ending ............................ 23
polyglossia .............................. 4, 12
polyglossia (package) ............... 4, 12, 28
post (property) ......................... 4
\printacronyms ....................... 1, 18 f., 28 f.
\ProvideAcroCommand ............... 29, 34
\ProvideAcroEnding .................. 33 ff.

R
\RenewAcroCommand .................. 29
reset-at-barriers .................... 28
Reutenauer, Arthur ................. 4, 12

S
short (property) ....................... 2 f., 5
short-format ............................ 22 f.
short-format (property) .......... 4, 6
short-indefinite (property) ...... 3, 12
short-plural (property) .......... 3, 9
short-plural-form (property) . 3
single .................................. 1, 4, 14, 20 f.
single .................................. 4
single-form ........................... 20, 23
single-format (property) ...... 5
sort .................................... 1, 18, 21
sort (property) ....................... 4 f., 18
strict ................................ 21
SW ...................................... 10

T
tabu ...................................... 37
tabu (package) ....................... 37
Talbot, Nicola L.C. ................. 13
ST ........................................ 6
The LaTeX Project Team ............ 35
tooltip ................................ 5, 17, 22
tooltip (property) .................. 5, 17
tooltip-cmd ............................. 5, 17, 22
translations .......................... 2
translations (package) .......... 2

U
uc-cmd .................................. 13, 22
UFO ....................................... 13
uppercase-short .................... 7 f., 13, 22
use-barriers .......................... 28

W
Wikipedia ............................... 6, 14 f., 24

X
xparse (package) .................... 2, 29 f.
xspace .................................. 21
xtemplate (package) .............. 2, 35

Z
zref-aps-page (package) ....... 2