ACRO

v3.0alpha-1 2020/04/07

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

! Hi and thanks that you are testing v3.0 of ACRO before it is released to CTAN. If you want to test the new version use `\usepackage[version=3]{acro}`. With `version=2` or no option at all you get the old version of acro. Using `\usepackage[version=3,deprecation]{acro}` is supposed to give as much meaningful warnings and errors as possible.

Part I.
Get started with ACRO

1. Licence and requirements

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

2. ACRO 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 properties.

In the document acronyms are used with these commands:

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

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

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

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

Let's say you defined CD as follows:

```
\DeclareAcronym{cd}{
  short = CD ,
  long  = compact disc
}
```

Then the usage is

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

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

3. Setting options

3.1. Load-time options

ACRO knows only a few set of load-time options which can be used as argument to `\usepackage`. To be more precise it knows only one such option:

`upgrade`
: When this option is used ACRO tries to give as much helpful and meaningful warning or error messages when a deprecated or removed command or setup is used. This is especially useful if you are upgrading from version 2.

3.2. Setup command

All options of ACRO that have *not* been mentioned in section 3.1 have to be set up either with this command

`\acsetup{`⟨*options*⟩`}`
: or as option to other commands. If this is possible then it is described when the corresponding commands are explained. Options usually follow a key/value syntax like and are always described in the following way:

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

`option = {`⟨*value*⟩`}` Initial: `preset`
: An option where a value can be given. The pre-set value is given to the right.

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

`option = true|false`
: A boolean option.

`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):

```
\acsetup{
  make-links = true , % boolean
  index ,             % boolean
  format = \emph ,    % standard
  list / local ,      % boolean option of the list module
  list/display = all  % choice option of the list module
}
```

Part II.
Comprehensive description of creation and usage of acronyms

4. 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 leeds to an error message.

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

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

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

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

4.1. 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 11 on page 16 for details.

`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 8 on page 12.

`sort` = {⟨*text*⟩} if unused then equal to `short`
If used the acronym will be sorted according to this property instead of its ID.

`class` = {⟨*csv list*⟩} (initially empty)
The class(es) the acronym belongs to.

`cite` = [⟨*prenote*⟩][⟨*postnote*⟩]{⟨*citation keys*⟩} (initially empty)
A citation that is printed to the acronym according to an option explained later.

`index` = {⟨*text*⟩}
Default This property allows to overwrite the automatic index entry with an arbitrary one. See section 15.2 on page 19 for details.

4.2. Properties related to plural and indefinite forms

`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.

`short-indefinite` = {⟨*text*⟩} Initial: `a`
Indefinite article for the short form.

`long-indefinite` = {⟨*text*⟩} Initial: `a`
Indefinite article for the long form.

`alt-indefinite` = {⟨*text*⟩} Initial: `a`
Indefinite article for the alternative form.

4.3. Properties related to formatting

`format` = {⟨TEX *code*⟩} (initially empty)
The format used for both short and long form of the acronym.

`short-format` = {⟨TEX *code*⟩} if unused then equal to `format`
The format used for the short form of the acronym.

`long-format` = {⟨TEX *code*⟩} if unused then equal to `format`
The format used for the long form of the acronym.

`alt-format` = {⟨TEX *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` = {⟨TEX *code*⟩} (initially empty)
The format used for the additional information of the acronym.

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

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

`first-style` = `long-short|short-long|short|long|footnote` (initially empty)
The style of the first appearance of the acronym, see also section 7 on page 11.

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

4.4. Properties related to the created PDF file

`pdfstring` = {⟨*pdfstring*⟩} if unused then equal to `short`
Used as PDF string replacement in bookmarks when used together with the hyperref [OR19] or the bookmark package [Obe19].

`pdfcomment` = {⟨*text*⟩}
Sets a tooltip description for an acronym. For actually getting tooltips you also need an appropriate setting of the options `pdfcomment/cmd` and `pdfcomment/use`, see also section 18.2 on page 19.

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

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

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

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

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

`single-acc` = {⟨*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.

4.5. Futher properties

`list` = {⟨*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.

`foreign-babel` = {⟨*language*⟩} (initially empty)
The babel [Bra19] or polyglossia [Cha19] 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. See section 11 on page 16 for details.

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 ID or (if set) by the value of the sort property. This property allows to set an individual sorting option for the index. See section 15.2 on page 19 for details

5. 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.

\acrocommand*[⟨options⟩]{⟨id⟩}
This is the general syntax of all of the commands listed below. The star and the optional argument is left way for the sake of readability.

\ac{⟨id⟩} \Ac{⟨id⟩} \acp{⟨id⟩} \Acp{⟨id⟩} \iac{⟨id⟩} \Iac{⟨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. The commands \acp and \Acp, resp., print the corresponding plural forms. The commands \iac and \Iac, resp., print indefinite forms.

\acs{⟨id⟩} \Acs{⟨id⟩} \acsp{⟨id⟩} \Acsp{⟨id⟩} \iacs{⟨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 \Acsp, resp., print the corresponding plural forms. The commands \iacs and \Iacs, resp., print indefinite forms.

\acl{⟨id⟩} \Acl{⟨id⟩} \aclp{⟨id⟩} \Aclp{⟨id⟩} \iacl{⟨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 \acfp and \Acfp, resp., print the corresponding plural forms. The commands \iacf and \Iacf, resp., print indefinite forms.

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

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

Then typical outputs look like this:

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

unidentified flying object (unbekanntes Flugobjekt, UFO)
a UFO
an unidentified flying object
An unidentified flying object (unbekanntes Flugobjekt, UFO)
unidentified flying objects (unbekannte Flugobjekte, UFOs)

6. Alternative short forms

Sometimes expressions have two different short forms. And example might be JPEG which also often is JPG. This is what the property `alt` is there for.

`alt` = {⟨*text*⟩}
Alternative short form.

Let's define JPEG:

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

And let's see how to use it:

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

Joint Photographic Experts Group (JPEG or JPG)
JPEG
JPG

As you can see the full form shows both short forms of the acronym. This could be changed by altering the template for the full form, see section 19 on page 19 and section 7. The alternative form is also printed in the list of acronyms, see section A on page 21. This can also be changed by altering the template for the list, again see section 19.

7. 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 5 on page 9) 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:

`first-style` = `long-short|short-long|short|long|footnote` Initial: `long-short`
The style of the first appearance of the acronym. This options sets the appearance for all acronyms. Available options in reality are the names of all defined templates of the type `acronym`. All pre-defined templates can be found in section 19.1 on page 19.

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

`first-style` = `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:

```
\acf[first-style=long-short]{cd} \\
\acf[first-style=short-long]{cd} \\
\acf[first-style=footnote]{cd} \\
\acf[first-style=long]{cd} \\
\acf[first-style=short]{cd}
```

compact disc (CD)
CD (compact disc)
CD[a]
compact disc
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. This usually requires at least two compilations.

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

and output like this:

```
\ac{etc}, \ac{etc} \ac{etc}.
```

et cetera, *etc. etc.*

The command `\acdot` is explained in section 14 on page 19. Basically it checks if a dot follows and outputs a dot if not.

8. 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 5 on page 9), then the single appearance of the acronym is printed.[2]

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

`single` = `true`|`false`|⟨*number*⟩ Initial: `false`

This option determines wether 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 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 19.1 on page 19.

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

`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.

`single-format` = {⟨*TEX code*⟩} if unused then equal to `long-format`

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

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

The style of the single appearance of the acronym.

2. This usually requires at least two compilations.

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
}
```

And it is used only once in this manual[3]:

```
\ac{png}
```

PNG

Please be aware that `\acf` would still print the full form, of course.

9. Plural forms and other endings

9.1. 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.[4] This is why ACRO has quite a number of different 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.

3. You will find it in the list of acronyms in section A nonetheless as this document does `list/display = {all}`.
4. German is full of such examples.

`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:

```
\DeclareAcronym{sw}{
  short = SW ,
  long = Sammelwerk ,
  long-plural = e
}
\DeclareAcronym{MP}{
  short = MP ,
  long  = Member of Parliament ,
  long-plural-form = Members of Parliament
}
```

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:

```
\acfp{sw} \par
\acfp{MP}
```

Sammelwerke (SWs)
Members of Parliament (MPs)

9.2. Other endings

There are other such concepts which is why ACRO generalizes the concept of endings.

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

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

```
\DeclareAcroEnding{plural}{s}{s}
```

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

short-⟨*foo*⟩-ending = {⟨*value*⟩}	Initial: ⟨*x*⟩
long-⟨*foo*⟩-ending = {⟨*value*⟩}	Initial: ⟨*y*⟩

and these properties

short-⟨*foo*⟩ = {⟨*value*⟩}	Initial: ⟨*x*⟩
short-⟨*foo*⟩-form = {⟨*value*⟩}	(initially empty)
alt-⟨*foo*⟩ = {⟨*value*⟩}	Initial: ⟨*x*⟩
alt-⟨*foo*⟩-form = {⟨*value*⟩}	(initially empty)
long-⟨*foo*⟩ = {⟨*value*⟩}	Initial: ⟨*y*⟩
long-⟨*foo*⟩-form = {⟨*value*⟩}	(initially empty)
foreign-⟨*foo*⟩ = {⟨*value*⟩}	Initial: ⟨*y*⟩
foreign-⟨*foo*⟩-form = {⟨*value*⟩}	(initially empty)
single-⟨*foo*⟩ = {⟨*value*⟩}	Initial: ⟨*y*⟩
single-⟨*foo*⟩-form = {⟨*value*⟩}	(initially empty)
extra-⟨*foo*⟩ = {⟨*value*⟩}	Initial: ⟨*y*⟩
extra-⟨*foo*⟩-form = {⟨*value*⟩}	(initially empty)

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

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

Section 20 on page 19 has an example of how this can be used to define a possessive ending and commands that make use of them like this:

```
1 \acfg{MP}                    Member's of Parliament (MP's)
```

10. Indefinite forms

Indefinite forms can be a problem if the short and the long form of acronyms have different indefinite articles.[5]

5. This may very well be a language specific issue.

```
\acreset{ufo}%
a \ac{ufo} \par
an \ac{ufo}
```

a unidentified flying object (unbekanntes Flugobjekt, UFO)
an UFO

And what good would it be to uase a package like ACRO if you have to keep track of of and second uses, anyway? This is why UFO should be defined like we did on page 10. We then can just use the dedicated commands and let them decide for us:

```
\acreset{ufo}%
\iac{ufo} \par
\iac{ufo}
```

an unidentified flying object (unbekanntes Flugobjekt, UFO)
a UFO

The commands which also output the indefinite article all start with an "i" and have all been described in section 5 on page 9 already: `\iac`, `\Iac`, `\iacs`, `\Iacs`, `\iacl`, `\Iacl`, `\iaca`, `\Iaca`, `\iacf`, and `\Iacf`.

11. 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 11 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` = {⟨*TEX code*⟩} (initially empty)
The format used for the foreign form of the acronym.

`foreign-babel` = {⟨*language*⟩} (initially empty)
The babel [Bra19] or polyglossia [Cha19] 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:

`locale/display` = `true`|`false` Initial: `false`

This options determines wether 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.

`locale/format` = {⟨*code*⟩} Initial: `\em\text_titlecase_first:n`

Determines how said language is formatted when printed. The last command in ⟨*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:

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

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

```
\ac{ecu} \par
\acsetup{locale/display,locale/format=\emph}
\acf{ecu}
```

Steuergerät (Electronic Control Unit, ECU)
Steuergerät (*englisch*: Electronic Control Unit, ECU)

The property `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 ACRO is able to deduce the language used for the "locale" field by itself:

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

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

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

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

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

Part IV.
Appendix

A. Examples

B. Acronyms

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

CD compact disc 3, 11

CTAN Comprehensive TEX Archive Network 2

e.g. for example (Latin: *exempli gratia*) 17

ECU Steuergerät (*Englisch*: Electronic Control Unit) 17

etc. *et cetera* 12

ID identification string 5f., 9

JPEG/JPG Joint Photographic Experts Group 10

LA Los Angeles

LPPL LATEX Project Public License

MP Member of Parliament . . . . . 14f.

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

NY New York

PDF Portable Document Format 8

PNG Portable Network Graphics 13

TEX.sx TEX StackExchange

UFO unidentified flying object (*German*: unbekanntes Flugobjekt) . . . . . 10, 15f.

C. References

[Bra19] Johannes Braams. *babel*. Version 3.33. July 19, 2019.
URL: https://www.ctan.org/pkg/babel/.

[Cha19] François Charette. *polyglossia*. Version 1.44. Apr. 4, 2019.
URL: https://www.ctan.org/pkg/polyglossia/.

[Obe19] Heiko Oberdiek. *bookmark*. Version 1.28. Dec. 3, 2019.
URL: https://www.ctan.org/pkg/bookmark/.

[OR19] Heiko Oberdiek and Sebastian Rahtz. *hyperref*. Version 6.88i. Sept. 12, 2019.
URL: https://www.ctan.org/pkg/hyperref/.

D. Index