

# L<sup>A</sup>T<sub>E</sub>X Package **contract**\*

Markus Kohm

Version v0.92 of 2026-04-23

The **contract** package is intended for cautelar jurisprudence. It is intended to provide flexible help for lawyers and notaries in drafting contracts, statutes and legal commentaries. It has been developed in cooperation with Dr Alexander Willand, and is still in the process of development.

Package **contract** is a **KOMA-Script** spin-off. It has been released from 2011 till 2023 as **scrjura**. With **KOMA-Script** 3.42 **scrjura** is removed from **KOMA-Script** and released as new package **contract**.

## Contents

<b>User Manual</b>	<b>2</b>
<b>1 Package Loading and Option Setting</b>	<b>3</b>
<b>2 Changing the Fonts of Elements</b>	<b>3</b>
<b>3 Clauses in the Table of Contents</b>	<b>4</b>
<b>4 Environment for Contracts</b>	<b>5</b>
4.1 Clauses . . . . .	5
4.2 Paragraphs . . . . .	7
4.3 Sentences . . . . .	9
<b>5 Cross References</b>	<b>9</b>
<b>6 Additional Contract Environments</b>	<b>12</b>
<b>7 Support for Different Languages</b>	<b>12</b>
<b>8 A Detailed Example</b>	<b>14</b>
<b>9 From scrjura to contract</b>	<b>18</b>
<b>10 State of Development</b>	<b>20</b>

---

\*The repository of this package can be found at <https://codeberg.org/komascript/latex-contract.git> where you also should report issues.

<b>Implementation</b>	<b>20</b>
11 Cooperation with <code>hyperref</code>	20
12 Prerequisites	20
13 Options	21
14 Contracts, Clauses, Paragraphs and Sentences	26
15 Entry to Table of Contents	36
16 Numbering of Paragraphs and Sentences	37
17 Referencing	39
18 Language Dependent Names	45
19 Using Values from last <code>LaTeX</code> Run	46
<b>References</b>	<b>47</b>
<b>Change History</b>	<b>47</b>
<b>Index</b>	<b>51</b>

## List of Tables

1	Available properties for the optional argument of <code>\Clause</code> and <code>\SubClause</code>	6
2	Available values for the <code>clausemark</code> option to activate running heads	7
3	Available values for the <code>ref</code> option to configure the cross reference format	11
4	Example outputs of the <code>ref</code> -independent cross reference commands	11
5	Properties provided by <code>\DeclareNewJuraEnvironment</code> for new contract environments	13
6	Meanings and English defaults of language-dependent terms	14

## List of Figures

1	The four pages of the CfCH example of section 8	19
---	---	----

# User Manual

If you want to write a contract, the articles of association for a company or an association, a law, or a legal commentary, the package `contract` will provide typographical support. Although `contract` is intended to provide general help for legal documents, the `contract` is the central element of the package. Particular attention is paid to clauses, titles, and numbered

provisions — if there are several of them in a clause —, numbered sentences, entries in the table of contents, and cross references according to German standards.

The package has been developed in cooperation with Dr Alexander Willand of Karlsruhe. Many of its features go back to constructive inquiries from Prof Heiner Richter of the Hochschule Stralsund University of Applied Sciences.

Some of you may search for the German user manual formally available in [Koh23a]. The author's acute overload has resulted in this no longer being freely available. However, the corresponding chapter in [Koh20a] and [Koh20b] is still accessible and applicable with appropriate adjustments to the package name.

## 1 Package Loading and Option Setting

You can load the package as common using:

```
\usepackage[<options>]{contract} .
```

In case of package `contract` the *<options>* are *<key>=<value>* options.

**Note:** In case of using package `contract` with package `hyperref` you should load `hyperref` always after `contract`.

`\contractSetup` To change options after loading the package, you should use:

```
\contractSetup{<options>}
```

with *<options>* is a comma separated list of *<key>=<value>* options as explained before.

**Note:** Currently the options are still implemented using the internal `KOMA-Script` package `scrkbase`<sup>1</sup>. Therefore you could also setup *<options>* using the `KOMA-Script` commands `\KOMAOPTIONS` or `\KOMAoption`, which are described in [Koh23b]. Moreover, because of the implementation, currently `\contractSetup` could also be used to setup other `KOMA-Script` options not related to package `contract`. However, you should avoid this, because it will fail, when the implementation will be changed.

## 2 Changing the Fonts of Elements

Currently package `contract` still uses the font selection features of the internal `KOMA-Script` package `scrkbase`. So the commands `\setkomafont`, `\addtokomafont`, and `\usekomafont`, which are described in [Koh23b], can be used to change the fonts of the following elements:

**Clause:**

Alias for *<environment>.Clause* within any contract environment, e. g., `contract.Clause` within environment `contract`. If no corresponding element is defined, `contract.Clause` is used.

`contract.Clause:` (Default: `\sffamily\bfseries\large`)

The heading of a `\Clause` within the environment `contract`.

---

<sup>1</sup>You need at least `KOMA-Script` v3.39 to have `scrkbase` and all other `KOMA-Script` packages required by `scrkbase` and `contract` installed. If you use the recommended installation with the package manager of your  $\text{\TeX}$  distribution this version requirement should be fulfilled by installing the `KOMA-Script` package or by updating an already installed `KOMA-Script`.

`<environment>.Clause:` (Default: *none*)

The heading of a `\Clause` within environment `<environment>`, which has been defined using `\DeclareNewJuraEnvironment`, if the font has been setup using property `ClauseFont` or the element has been defined explicitly.

`parnumber:` (Default: *empty*)

The paragraph number within a contract environment.

`sentencenumber:` (Default: *empty*)

The sentence number printed by `\Sentence`.

**Note:** There are plans to no longer use package `scrkbase`, but maybe package `scxextend` instead, if `scrjura` is used with a non-KOMA-Script class. So the commands `\setkomafont`, `\addtokomafont` and `\usekomafont` would still be available in future.

### 3 Clauses in the Table of Contents

The headings of clauses can also be added automatically to the table of contents, if desired. For this the package uses command `\DeclareTOCStyleEntry` of KOMA-Script package `tocbasic` to define an entry level named `cpar`. Usage of package `tocbasic` also means, that you should not use another package to configure the Table of Contents, e.g., `tocloft`, `tocstyle` etc.

`juratotoc (opt.)`

Clauses are shown in the table of contents only if their level number is less than or equal to the `tocdepth` counter. By default, the level number is `\maxdimen`, which is also used if the option is switched off using `juratotoc=false`. Because the `tocdepth` counter usually has a one-digit value, clause entries are therefore not normally displayed in the table of contents.

If you switch on the option using the `juratotoc=true`, the level number 2 is used so that clauses are shown in the table of contents on the same level as subsections. For the default setting of `tocdepth`, clauses are then shown in all KOMA-Script classes or standard classes.

You can also use `juratotoc=<integer>` to use `<integer>` instead of `\maxdimen` or 2 as level number.

Internally usage of this option results in a call of

```
\DeclareTOCStyleEntry[level=<integer>]{default}{cpar}
```

respectively

```
\DeclareTOCStyleEntry[level=2]{default}{cpar}
```

respectively

```
\DeclareTOCStyleEntry[level=\maxdimen]{default}{cpar} .
```

These options determine the indentation and spacing for clause entries in the table of contents. Any valid `<length>` can be assigned. The defaults are the same as for subsection entries in `scrartcl`.

Internally, usage of these options results in calls to:

```
\DeclareTOCStyleEntry[indent=<length>]{default}{cpar}
```

respective

```
\DeclareTOCStyleEntry[numwidth=<length>]{default}{cpar} .
```

`juratocindent (opt.)`  
`juratocnumberwidth (opt.)`

## 4 Environment for Contracts

The essential mechanisms of package `contract` are available only inside contract environments, either the predefined `contract` or any other environment defined with command `\DeclareNewJuraEnvironment`.

`contract` Currently, this is the one and only predefined environment for `contract`. Using it activates automatic numbering of paragraphs and the `\Clause` and `\SubClause` commands, which will be documented below, are given concrete form.

The `contract` environment must not be nested within itself. Within a document, however, you can use the environment several times. The clauses within these environments are treated as if they were within a single environment. As a result, ending the environment really only temporarily interrupts it, and the old environment is continued by the beginning of a new environment. However, you cannot end the environment within a clause. If you want instead print several contracts, you would need to define several contract environments using command `\DeclareNewJuraEnvironment`.

`contract (opt.)` The whole document becomes a contract if you use this option while loading the package with:

```
\usepackage[contract]{contract}
```

or as a global option with `\documentclass`. The document then behaves exactly as if it would contain one `contract` environment.

### 4.1 Clauses

Clauses<sup>2</sup> in a legal sense are defined in package `contract` only within contracts, that is inside the `contract` environment or other environments declared with `\DeclareNewJuraEnvironment`.

`\Clause` Each clause starts either with `\Clause[⟨property list⟩]` or `\SubClause[⟨property list⟩]`. The  
`\SubClause` optional argument `⟨property list⟩` is a comma separated list of `⟨key⟩=⟨value⟩`.

These are the most important commands inside of a contract. Without using any additional `⟨key⟩`, `\Clause` creates the heading of a clause, which consists of the sign “§”, followed by its number. In contrast, `\SubClause` creates the heading of a clause with the last number used by `\Clause` and adds a lower-case letter. `\SubClause` is mainly intended for cases where an act or a contract is amended and not only are clauses changed or deleted but new clauses are inserted between existing ones without completely changing the numbering.

Both commands accept a comma-separated list of `⟨key⟩=⟨value⟩` properties and also some `⟨key⟩`s without value. An overview of the available properties is shown in [Table 1](#). The most important of them will be discussed in more detail.

By default, a skip of two lines is inserted before the heading and a skip of one line afterwards. You can change the size of these skips with the `preskip` and `postskip` properties. The new values apply not only to the current clause but from the current clause until the end of the current contract environment. You can also make the appropriate settings in advance with

```
\setkeys{contract}{preskip=⟨skip⟩,postskip=⟨skip⟩}
```

<sup>2</sup>In English, a “clause” in a legal document is a section, paragraph, or phrase that relates to a particular point. Although it is common in English to also use the terms “article” or “section” for what we here call a “clause”, we use the latter term throughout to avoid confusion with the `article` class and the `\section` and `\paragraph` sectioning divisions of most document classes.

Table 1: Available properties for the optional argument of `\Clause` and `\SubClause`

<code>dummy</code>	The heading will not be printed but is counted in the automatic numbering.
<code>head=&lt;running head&gt;</code>	If running heads are active, this <i>&lt;running head&gt;</i> is used instead of the clause <i>&lt;title&gt;</i> .
<code>nohead</code>	The running head stays unchanged.
<code>notocentry</code>	Does not make an entry into the table of contents.
<code>number=&lt;number&gt;</code>	Uses <i>&lt;number&gt;</i> for the output of the clause number.
<code>preskip=&lt;skip&gt;</code>	Changes the vertical <i>&lt;skip&gt;</i> before the clause heading.
<code>postskip=&lt;skip&gt;</code>	Changes the vertical <i>&lt;skip&gt;</i> after the clause heading.
<code>title=&lt;title&gt;</code>	The clause <i>&lt;title&gt;</i> will be printed in addition to the clause number. This is also used as the default for the <i>&lt;running head&gt;</i> and the <i>&lt;entry&gt;</i> in the table of contents.
<code>tocentry=&lt;entry&gt;</code>	Regardless of the clause <i>&lt;title&gt;</i> , an <i>&lt;entry&gt;</i> into the table of contents will be made, if such entries are activated (see option <code>juratotoc</code> ).

regardless of the specific clause and outside of a contract environment. You can also set these options inside the preamble after loading `contract`, but you cannot set them while loading the package or by using `\contractSetup`.

By default, clause headings use the font style `\sffamily\bfseries\large`. See section 2 for information about how to change the font for element `contract.Clause`.

With the `title`, `head`, and `tocentry` property, you can title a clause in addition to the number. You should enclose the *<value>* of each property inside curly brackets. Otherwise, for example, commas which are meant to be part of the *<value>* will be confused with the delimiters between different properties of the *<property list>*. Empty values for `head` and `tocentry` cause empty entries. If you want to avoid an entry, use the `nohead` resp. `notocentry` property.

Instead of consecutive numbers, you can also set a clause number manually with the `number` property. However, this does not affect the numbers of the subsequent clauses. Empty numbers are not possible. Fragile commands inside `number` have to be protected with `\protect`. You should use only numbers and letters as a `number`.

With the `dummy` property, you can suppress the output of the whole heading of a clause. The automatic numbering, however, will still count this clause. In this way, you can skip an automatically numbered clause with

`\Clause[dummy]`

in case the clause corresponding clause has been deleted in a later version of a contract.

Note that the `dummy` property only accepts the *<value>*s `true` and `false`. All other *<value>*s are usually ignored, but can lead to an error message in the worst case scenario.

`\Clauseformat`

As already mentioned, clauses and subclauses are normally numbered. The number is formatted with the help of the `\Clauseformat` command, which expects the *<number>* as the only argument. The default is the following:

`\newcommand*{\Clauseformat}[1]{\S #1}`

Table 2: Available values for the `clausemark` option to activate running heads

<code>both</code>	Clauses generate left and right marks for running heads, if the document provides automatic running heads.
<code>false</code> , <code>off</code> , <code>no</code>	Clauses do not generate marks for running heads and therefore do not change running heads.
<code>forceboth</code>	Clauses use <code>\markboth</code> to generate left and right marks for running heads even if the document does not provide automatic running heads for the current page style.
<code>forceright</code>	Clauses use <code>\markright</code> to generate right marks for running heads even if the document does not provide automatic running heads for the current page style.
<code>right</code>	Clauses generate right marks for running heads, if the document provides automatic running heads.

This produces the section mark, `\S` (§), followed by a non-breaking space and the number. If you redefine this command, be sure it remains expandable.

`juratitlepagebreak` (*opt.*)

Usually, page breaks are prohibited within heading of all kinds. However, some lawyers require page breaks within clause headings. You can allow such a break by using option `juratitlepagebreak` or `juratitlepagebreak=<boolean>`. Boolean values of `true` or `false` can be used to toggle the option on or off. Using the option without a value is the same as using `true`. The option can be used as an optional argument of `\documentclass`, `\usepackage` when loading package `contract`, or as an argument of `\contractSetup`.

`clausemark` (*opt.*)

Since clauses are a subordinate structure with independent numbering, they do not produce running heads by default. You can, however, create running heads with various settings using:

`clausemark=<value>`

as an optional argument of `\documentclass`, `\usepackage` when loading package `contract`, or as an argument of `\contractSetup`. You can find the available `<value>`s and their meanings in Table 2.

## 4.2 Paragraphs

Within clauses, `contract` usually numbers paragraphs automatically. With this, the paragraphs provide a powerful structuring element, similar to `\paragraph` or `\subparagraph` in normal documents. For this reason, contracts usually use a vertical skip between paragraphs. The `contract` package does not provide its own mechanism for this. Instead, you should use the `parskip` option of the `KOMA-Script` classes. If you do not use a `KOMA-Script` class, see the documentation of the used class or package `parskip` [Mit21].

`parnumber` (*opt.*)

The option can be used as an optional argument of `\documentclass`, `\usepackage` when loading package `contract`, or as an argument of `\contractSetup`.

The default numbering of paragraphs is `parnumber=auto` and `parnumber=true`. Sometimes you may need to disable the automatic numbering. You can do this with `parnumber=false`. In this case, only the sentence numbering is reset.

To implement this option, it has been necessary to hook into the paragraph-building mechanism of L<sup>A</sup>T<sub>E</sub>X. In some rare cases, this can have a negative effect. If so, you can undo the change with `parnumber=manual`. On the other hand, L<sup>A</sup>T<sub>E</sub>X itself sometimes undoes the change. In those cases you can activate it again with `parnumber=auto`.

Clauses that consist of a single paragraph do not automatically receive a paragraph number. For this to work, there must not be two clauses with an identical number in a document. However should you ever need such numbering, you should switch to another contract environment (see `\DeclareNewJuraEnvironment`). Note that the number of paragraphs in a clause is not available before the end of the clause. Therefore you need a least two L<sup>A</sup>T<sub>E</sub>X runs before the automatic paragraph numbering is correct.

`par (cnt.)` For numbering the paragraphs inside a clause we use the `par` counter. The output of `\thepar` will display an Arabic number, because the default is `\arabic{par}`. `\parformat` provides the format, which is `\thepar` in rounded brackets. When numbering a paragraph manually, you should also use `\parformat`. It makes sense to call `\parformat` with a subsequent `\parformatseparation`, or at least a `\nobreakspace` or tilde.

With automatic numbering, `\parformat` is followed by `\parformatseparation`, which currently consists of `\nobreakspace`, the non-breakable space.

The paragraph number is usually printed using the currently active font. See [section 2](#) for information how to change the font of the `parnumber` element.

Note: `contract` currently assumes internally that `\thepar` is an Arabic number. Therefore you should definitely not redefine it!

`\withoutparnumber` If the paragraph number is not printed, `contract` executes the `\withoutparnumber` command at the beginning of the new paragraph. The initial definition of this command is empty. This means it is a kind of dummy command that does nothing. It has been implemented because of a user request. Most users can ignore this command.

`\ellipsispar` Sometimes — particularly in comparative commentaries — it is desirable to omit paragraphs but to mark the omission. Those omitted paragraphs should be taken into account by the paragraph counter. The package `contract` provides the command `\ellipsispar` to do this.

By default, `\ellipsispar` omits precisely one paragraph. Using the optional argument of

`\ellipsispar[⟨number⟩]`

you can omit multiple paragraphs. In any case, the output shows just one unnumbered paragraph, which consists only of the ellipsis defined by `\parellipsis`. The automatic numbering of paragraphs takes the `⟨number⟩` of omitted paragraphs into account.

**Example:** Suppose you are writing a comment on the German<sup>3</sup> penal code, but only on paragraph 3 of § 2. Nevertheless, you’d like to indicate the omission indirectly. You can do this with:

```
\documentclass[parskip=half]{scrartcl}
\usepackage{contract}
\begin{document}
\begin{contract}
\Clause{title={Temporal application},number=2}
```

<sup>3</sup>Please remember, this translation does not refer to an existing law but is only an example of how you might realise such a commentary with `contract`.



```
\ellipsispar[2]
```

```
If the law that applies at the time the criminal act is
committed is changed before the verdict, then the most
lenient law shall be applicable.
```

```
\ellipsispar[3]
\end{contract}
\end{document}
```

To see the result, just give it a try.

The ellipsis is by default `\textellipsis`, if such a command is defined. If not, `\dots` is used. You can redefine `\parellipsis` at any time with `\renewcommand`.

### 4.3 Sentences

Paragraphs in contracts consist of one or more sentences, some of which may also be numbered. However, as automatic numbering is cumbersome and error-prone, it has not yet been implemented in `contract`. Semi-automatic numbering, however, is supported.

```
sentence (cnt.)
\thesentence
\sentencenumberformat
\Sentence
```

Manual numbering of sentences is done with the `\Sentence` command. It adds one to the `sentence` counter. By default, `\sentencenumberformat` prints `\thesentence` as an Arabic number in superscript.

The sentence number is usually printed using the currently active font. See [section 2](#) for information how to change the font of the `sentence` element.

Using `babel` offers an easy way to define a shorthand for `\Sentence`:

```
\usesshorthands{' }
\defineshorthand{'S}{\Sentence\ignorespaces}
```

With this definition, any space after 'S will be ignored. You can even use the dot as an abbreviation for a dot and a new sentence number:

```
\defineshorthand{'.'}{. \Sentence\ignorespaces}
```

For details regarding `\usesshorthands` and `\defineshorthand`, please consult the manual of the `babel` package (see [\[BB24\]](#)). You can find an example of their application in [section 8, page 14](#).

## 5 Cross References

The conventional mechanism to set cross references using `\label`, `\ref`, and `\pageref` does not suffice for clauses, paragraphs, and sentences. Therefore `contract` provides additional commands.

```
\ref      The commands
\refL     \ref{\label}
\refS     \refL{\label}
\refN     \refS{\label}
          \refN{\label}
```

give a full reference to clause, paragraph and sentence. `\refL` is a long text, `\refS` a short text, and `\refN` an abbreviated, numeric form. `\ref` defaults to `\refL`.

```
\refClause  The commands
\refClauseN
```

```

\refClause{\label{}}
\refClauseN{\label{}}

```

reference a clause without displaying the paragraph or sentences. `\refClause` puts a section mark (§) in front of the reference, while `\refClauseN` does not.

The commands

```

\refPar
\refParL
\refParS
\refParN

```

```

\refPar{\label{}}
\refParL{\label{}}
\refParS{\label{}}
\refParN[number format]{\label{}}

```

reference a paragraph of a clause. The differences between the forms correspond to the differences between `\refL`, `\refN` and `\refS`. A feature worth noting is the optional argument of `\refParN`. Usually the numeric reference to a paragraph uses a Roman number. You can, however, specify a different *number format* in the optional argument. This option primarily makes sense to use Arabic numbers. By default, `\refPar` is `\refParL`.

The commands

```

\refSentence
\refSentenceL
\refSentenceS
\refSentenceN

```

```

\refSentence{\label{}}
\refSentenceL{\label{}}
\refSentenceS{\label{}}
\refSentenceN{\label{}}

```

reference a sentence of a paragraph. Again, there is a long text form, a short text form, and a numerical form. By default, `\refSentence` is `\refSentenceL`.

The results of `\ref`, `\refPar`, and `\refSentence` depend on the option

```

ref=value

```

that can be used as an optional argument of `\documentclass`, `\usepackage` when loading package `contract`, or as an argument of `\contractSetup`. The default is `ref=long` and therefore `\refL`, `\refParL` and `\refSentenceL`. You can find the available *value*s for this option and their meaning in Table 3.

**Example:** Suppose you always want to reference paragraphs in the form “paragraph 1 in clause 1”. As there is no predefined command for this, you have to create your own definition from the available options. You can achieve this easily with:

```

\newcommand*{\refParM}[1]{%
  paragraph~\refParN[arabic]{#1}
  in clause~\refClauseN{#1}%
}

```

This new command can be used in the same way as `\refParL`.

You can find examples of results of the basic commands — this means the commands, that are independent from option `ref` — in Table 4.

Table 3: Available values for the `ref` option to configure the cross reference format of `\ref`, `\refPar`, and `\refSentence`

<code>long</code>	A combination of <code>parlong</code> and <code>sentencelong</code> .
<code>numeric</code>	A combination of <code>parnumeric</code> and <code>sentencenumeric</code> .
<code>clauseonly</code> , <code>onlyclause</code> , <code>ClauseOnly</code> , <code>OnlyClause</code>	A combination of <code>paroff</code> and <code>sentenceoff</code> . Note that <code>\refPar</code> and <code>\refSentence</code> produce empty results!
<code>parlong</code> , <code>longpar</code> , <code>ParL</code>	Paragraphs are referenced in long textual form.
<code>parnumeric</code> , <code>numericpar</code> , <code>ParN</code>	Paragraphs are referenced in simple numerical form.
<code>paroff</code> , <code>nopar</code>	Paragraphs have no reference. Note that <code>\refPar</code> produces an empty result!
<code>parshort</code> , <code>shortpar</code> , <code>ParS</code>	Paragraphs are referenced in short textual form.
<code>sentencelong</code> , <code>longsentence</code> , <code>SentenceL</code>	Sentences are referenced in long textual form.
<code>sentencenumeric</code> , <code>numeralsentence</code> , <code>SentenceN</code>	Sentences are referenced in simple numeric form.
<code>sentenceoff</code> , <code>nosentence</code>	Sentences have no reference. Note that <code>\refSentence</code> produces an empty result!
<code>senceshort</code> , <code>shortsentence</code> , <code>SentenceS</code>	Sentences are referenced in short textual form.
<code>short</code>	A combination of <code>parshort</code> and <code>senceshort</code> .

Table 4: Example outputs of the `ref`-independent cross reference commands

Command	Example output
<code>\refL{\label}</code>	§ 1 paragraph 1 sentence 1
<code>\refS{\label}</code>	§ 1 par. 1 sent. 1
<code>\refN{\label}</code>	§ 1 I 1.
<code>\refClause{\label}</code>	§ 1
<code>\refClauseN{\label}</code>	1
<code>\refParL{\label}</code>	paragraph 1
<code>\refParS{\label}</code>	par. 1
<code>\refParN{\label}</code>	I
<code>\refParN[arabic]{\label}</code>	1
<code>\refParN[roman]{\label}</code>	i
<code>\refSentenceL{\label}</code>	sentence 1
<code>\refSentenceS{\label}</code>	sent. 1
<code>\refSentenceN{\label}</code>	1.

## 6 Additional Contract Environments

Some users do not use `contract` to draft contracts or commentaries on individual laws but to examine different types of laws, which may not necessarily use the section prefix (§) before the title of each clause but perhaps something like “Art.” or “IAS”, and so forth. An independent counter is also required for each of these different clause types.

`\DeclareNewJuraEnvironment`

You can use:

```
\DeclareNewJuraEnvironment{environment}[\property list]
{\start commands}{end commands}
```

to define new and independent environments for contracts or other legal texts. The argument `<environment>` is the name of the new environment, of course. The `<start commands>` are commands which will be executed at the beginning of the environment, as if they were added directly after `\begin{environment}`. Correspondingly `<end commands>` will be executed at the end of the environment, as if added directly before `\end{environment}`. Without a `<property list>` the new environment behaves like the `contract` environment, but with its own counters. You can use several `<key>=<value>` properties as a comma-separated `<property list>`. See Table 5 for the currently supported `<options>`.

**Example:** To define the environment for articles we mentioned in the preface of this section, it is sufficient to write:

```
\DeclareNewJuraEnvironment{Article}[ClauseNumberFormat=Art.~]{}{}
```

If we are using a `KOMA-Script` class and want to separate the paragraphs in this environment with space instead of using paragraph indentation, we can use:

```
\DeclareNewJuraEnvironment{Article}[ClauseNumberFormat=Art.~]
{\KOMAOptions{parskip}}{}
```

In cross references, “Art.” will of course be used instead of “§”.

The new environment is used like `contract`:

```
\begin{Article}
\Clause{}
Human dignity is inviolable. To respect and protect people is a
duty of all state authority.
\end{Article}
```

## 7 Support for Different Languages

The `contract` package has been developed in cooperation with a German lawyer. Therefore it initially supported only the languages `german`, `ngerman`, `austrian`, and `naustrian`. Nevertheless, it has been designed to support common language packages like `babel`. Users can easily make changes by using `\providecaptionname`. If you have definitive information about the correct legal terms and conventions of a language, please contact the `KOMA-Script` author. Support for English has been added in this way, and so `contract` now also provides terms for the languages `english`, `american`, `british`, `canadian`, `USenglish`, and `UKenglish`.

`\parname`  
`\partshortname`  
`\sentencename`  
`\sentenceshortname`

These are the language-dependent terms used by `contract`. The meaning of the terms and their English defaults are shown in Table 6. The package itself defines them by using

Table 5: Properties provided by `\DeclareNewJuraEnvironment` for new contract environments

<code>Clause=⟨command⟩</code>	Defines the <code>⟨command⟩</code> to which the <code>\Clause</code> command is mapped within the environment. This <code>⟨command⟩</code> , like the one documented for <code>contract</code> , expects exactly one argument. To use it correctly requires advanced knowledge of the <code>contract</code> 's internal functioning. Furthermore, the requirements for the <code>⟨command⟩</code> may change in future versions. Therefore it is recommended not to use this option!
<code>ClauseFont=⟨commands⟩</code>	If this property is used, a new <code>⟨environment⟩.Clause</code> element is defined with the <code>⟨commands⟩</code> used as its default setting. If the element was previously defined as an alias, it will become an independent element instead. If it has already been defined as an independent element, the <code>⟨commands⟩</code> are used as new font settings. Please note the limitations for font settings in <a href="#">section 2</a> .
<code>SubClause=⟨command⟩</code>	Defines the <code>⟨command⟩</code> to which the <code>\SubClause</code> command is mapped within the environment. This <code>⟨command⟩</code> , like the one documented for <code>contract</code> , expects exactly one argument. To use it correctly requires advanced knowledge of the <code>contract</code> 's internal functioning. Furthermore, the requirements for the <code>⟨command⟩</code> may change in future versions. Therefore it is recommended not to use this property!
<code>Sentence=⟨command⟩</code>	Defines the <code>⟨command⟩</code> to which the <code>\Sentence</code> is mapped within the environment. This <code>⟨command⟩</code> must not have an argument. Typically it should add one to the <code>sentence</code> (using <code>\refstepcounter</code> ) counter and display it appropriately. It is particularly important to avoid adding unwanted spaces.
<code>ClauseNumberFormat=⟨command⟩</code>	Formats the numbers of clauses within the environment. The <code>⟨command⟩</code> should expect exactly one argument: the number of the clause. If the <code>⟨command⟩</code> implements a series of commands and the number is the last argument of a that series, you can directly use the series of commands as <code>⟨command⟩</code> .

Table 6: Meanings and English defaults of language-dependent terms, if not already defined

Command	Meaning	Default
<code>\parname</code>	long form “paragraph”	paragraph
<code>\parshortname</code>	short form “paragraph”	par.
<code>\sentencename</code>	long form “sentence”	sentence
<code>\sentenceshortname</code>	short form “sentence”	sent.

`\providecaptionname` inside `\begin{document}` only if other requirements have not already been met. If you use `contract` with an unsupported language, the package will throw an error.

## 8 A Detailed Example

You may remember the letter from the `scrlltr2` chapter of the `KOMA-Script` manual [Koh23b], in which a club member wanted to remind the board about an overdue meeting that was prescribed by the club’s by-laws. Such club by-laws are a kind of contract, and you can create them using `contract`.

```
\documentclass[fontsize=12pt,parskip=half]
{scrartcl}
```

We use class `scrartcl`. Because paragraph distance instead of paragraph indentation is usual in club by-laws, we load the class with option `parskip=half`.

```
\usepackage[british]{babel}
```

The club rules are in British English. Therefore we load the `babel` package with the `british` option too.

```
\usepackage[T1]{fontenc}
\usepackage{lmodern}
```

We make some default font settings. Earlier versions of the example also loaded the `textcomp` package here for an improved section mark (§). Since L<sup>A</sup>T<sub>E</sub>X 2020/02/01, however, the desired functionality is directly integrated in the L<sup>A</sup>T<sub>E</sub>X kernel.

```
\usepackage{enumerate}
```

Later in the document, we want lists numbered not with Arabic numbers but with lower-case letters. We can do this easily with the `enumerate` package. Alternatively, we could have used the `enumitem` package.

```
\usepackage[clausemark=forceboth,
    juratotoc,
    juratocnumberwidth=2.5em]
{contract}
\usesorthands{}
\defineshorthand{'S'}{\Sentence\ignorespaces}
\defineshorthand{'.'}{. \Sentence\ignorespaces}

\pagestyle{myheadings}
```

Now it is time for `contract`. The `clausemark=forceboth` option forces clauses to create left and right marks for the running head. On the other hand, we do not want `\section` to

change the marks for the running head. Therefore we use the `myheadings` page style. This page style generally does not provide automatic running heads.

Later, we also want a table of contents with the clauses. This can be achieved with the `juratotoc` option. Doing so we will see that the default width for these numbers is insufficient for the clause numbers in the table of contents. With `juratocnumberwidth=2.5em`, we reserve more space.

The definition of shorthands has already been explained in [section 4.3](#). In this example we do the same thing to simplify the input.

```
\begin{document}
```

It is time to begin the actual document.

```
\subject{By-Laws}
\title{CfCH}
\subtitle{Club for Club Hoppers}
\date{11.\,11.\,2011}
\maketitle
```

Like other documents, the by-laws have a title. We created it with the usual `KOMA-Script` commands.

```
\tableofcontents
```

As already mentioned, we want to create a table of contents.

```
\addsec{Preamble}
```

```
The club landscape in England is diverse. But we have
unfortunately been forced to conclude that it often
suffers seriously when dealing with seriousness.
```

Preambles are not unusual in club by-laws. Here we use `\addsec` to create one because we want it to have an entry in the table of contents.

```
\appendix
```

Here we use a small trick. The articles of the club by-laws should be numbered with upper-case letters instead of Arabic numbers, just as the appendix sections of an article using `scrartcl` are.

```
\section{Overview}
```

```
\begin{contract}
```

We begin the contract with the first article.

```
\Clause[title={Name, Legal Form, Headquarters}]
```

```
The name of this club shall be the ‘‘Club for Club
Hoppers’’ and is not registered in any club register.
```

```
’S The club is a non-economic, useless club’. It has no
headquarters because its members heads are in their
hindquarters.
```

```
The fiscal year is from March 31st through April 1st.
```

The first clause has a number and a title. We will do the same with all following clauses.

The first paragraph of the clause contains nothing unusual. Because it is not the only paragraph, every paragraph will be automatically preceded by a paragraph number. Note that the numbering the first paragraph requires at least two L<sup>A</sup>T<sub>E</sub>X runs. Since this is the case for the table of contents as well, this does not create any additional problems.

In the second paragraph we have two sentences. Here we can see the shorthands 'S and ' . in action. The first one only generates the sentence number. The second one generates not only the full stop but also the sentence number. With this, both sentences are numbered.

```
\Clause[title={Purpose of the Club}]
```

```
'S The club is pointless but not useless'. Rather,
it should put the serious handling of seriousness on a
sound footing.
```

```
For this purpose, the club members can
\begin{enumerate}[\qqquad a)]
\item pick their noses,
\item crack nuts,
\item such their thumbs.
\end{enumerate}
```

```
The club is selfish and stands by it.
```

```
The club has no financial means.\label{a:mittel}
```

The second clause: again this contains several paragraphs, some of which include several sentences. The second paragraph also has a numbered list. In the last paragraph, we set a label, because we want to reference it later.

```
\Clause[title={Club Officers}]
```

```
The club officers hold honorary positions.
```

```
'S If the club had resources (see \ref{a:mittel}), it
could afford a full-time manager'. Without the necessary
funds, this is not possible.
```

The third clause contains something special: a cross reference. Here we use the long form with clause, paragraph, and sentence. If we decided later that sentences should not be included in the reference, we could use the `ref=nosentence` option to set this globally.

```
\Clause[title={Club Hopper},dummy]
\label{p.maier}
```

Here we have a special kind of clause. In earlier versions of the club by-laws, this was a real clause, but it was later removed. However, the numbering of the following clauses should not be changed by removing this one. Therefore the `\Clause` statement has not been removed but supplemented by `dummy` property. With this, we also can set a label even though the clause will not be printed.

```
\end{contract}
```

```
\section{Membership}
```

```
\begin{contract}
```



Another article begins. To avoid problems with the paragraph numbering, we interrupt the `contract` environment.

```
\Clause[title={Types of Members},dummy]
```

The first clause of the next article also has been deleted.

```
\Clause[title={Becoming a Member}]
```

```
Everyone can purchase a membership from one of the
associations listed in \refClause{p.maier}.\label{a.preis}
```

```
'S To become a member, an informal application is
required'. This application should be submitted in green
ink on pink paper.
```

```
Membership applications cannot be rejected.
```

Here we have a real clause again. We cross reference one of the deleted clauses and also set a label.

```
\SubClause[title={Amendment to the Previous Clause}]
```

```
'S With the repeal of \refClause{p.maier},
\ref{a.preis} has become impractical'. In its place,
memberships can be inherited.
```

Once more, this is a special kind of clause. This time we have not removed a clause but added one without renumbering the following clauses. To do so, we use `\SubClause`. Therefore the clause number is the same like the previous one but with an appended “a”.

```
\Clause[title={Termination of Membership}]
```

```
'S Membership ends with one's life'. For non-living
members, membership does not end.
```

```
\Clause[title={General Meeting}]
```

```
A general meeting shall take place twice per year.
```

```
The interval between two general meetings shall be
no more than 6~months, 1~week, and 2~days.
```

```
The invitation to the next general meeting shall be sent
no earlier than 6~months from the previous general
meeting.
```

```
\SubClause[title={Amendment to the General Meeting}]
```

```
The general meeting may be held at the earliest 2~weeks after
the invitation is received.
\end{contract}
```

The other clauses of this article are very usual. You already know all the features used for them.

```
\section{Validity}
```

```

\begin{contract}
\Clause[title={Effective Date}]

These articles will enter into force on 11.\,11.\,2011 at
11:11~am.

'S If any provision of these by-laws is in conflict with
any other, the by-laws will be repealed on
11.\,11.\,2011 at 11:11~am and 11~seconds'. The club is
considered to be dissolved in this case.

\end{contract}

```

There follows another article no special features.

```
\end{document}
```

Then the L<sup>A</sup>T<sub>E</sub>X document ends. You can see first three pages in [Figure 1](#).

## 9 From scrjura to contract

If you have been using the scrjura package and are now switching to the contract package, there are a few things you need to be aware of:

- Instead of

```
\usepackage[options]{scrjura} ,
```

should now use

```
\usepackage[options]{contract} .
```

The same *options* are allowed as the scrjura package understands.

- Instead of using \KOMAOPTIONS or \KOMAOPTION to set *options* for the contract package, use \contractSetup.
- The \Clause and \SubClause commands now have an optional argument instead of a mandatory argument. This makes more sense because it is allowed to use \Clause or \SubClause without an argument. Instead of braces, the list of properties must be given in square brackets, for example:

```
\Clause[title={title of clause}]
```

Therefore, even more care should be taken to ensure that the values of each property are enclosed in braces. Otherwise, there may be problems not only with commas, but also with square brackets in the values.

- Check your code if you have used or redefined any internal macros as several of them have been renamed.

<p style="text-align: center;"><b>By-Laws</b></p> <p style="text-align: center;"><b>CfCH</b></p> <p style="text-align: center;"><b>Club for Club Hoppers</b></p> <p style="text-align: center;">11. 11. 2011</p> <p><b>Contents</b></p> <table> <tr> <td><b>Preamble</b></td> <td><b>1</b></td> </tr> <tr> <td><b>A. Overview</b></td> <td><b>2</b></td> </tr> <tr> <td>§ 1. Name, Legal Form, Headquarters</td> <td>2</td> </tr> <tr> <td>§ 2. Purpose of the Club</td> <td>2</td> </tr> <tr> <td>§ 3. Club Officers</td> <td>2</td> </tr> <tr> <td><b>B. Membership</b></td> <td><b>3</b></td> </tr> <tr> <td>§ 6. Becoming a Member</td> <td>3</td> </tr> <tr> <td>§ 6a. Amendment to the Previous Clause</td> <td>3</td> </tr> <tr> <td>§ 7. Termination of Membership</td> <td>3</td> </tr> <tr> <td>§ 8. General Meeting</td> <td>3</td> </tr> <tr> <td>§ 8a. Amendment to the General Meeting</td> <td>4</td> </tr> <tr> <td><b>C. Validity</b></td> <td><b>4</b></td> </tr> <tr> <td>§ 9. Effective Date</td> <td>4</td> </tr> </table> <p><b>Preamble</b></p> <p>The club landscape in England is diverse. But we have unfortunately been forced to conclude that it often suffers seriously when dealing with seriousness.</p> <p style="text-align: center;">1</p>	<b>Preamble</b>	<b>1</b>	<b>A. Overview</b>	<b>2</b>	§ 1. Name, Legal Form, Headquarters	2	§ 2. Purpose of the Club	2	§ 3. Club Officers	2	<b>B. Membership</b>	<b>3</b>	§ 6. Becoming a Member	3	§ 6a. Amendment to the Previous Clause	3	§ 7. Termination of Membership	3	§ 8. General Meeting	3	§ 8a. Amendment to the General Meeting	4	<b>C. Validity</b>	<b>4</b>	§ 9. Effective Date	4	<p style="text-align: center;">§ 1 Name, Legal Form, Headquarters</p> <p><b>A. Overview</b></p> <p><b>§ 1 Name, Legal Form, Headquarters</b></p> <p>(1) The name of this club shall be the "Club for Club Hoppers" and is not registered in any club register.</p> <p>(2) <sup>1</sup>The club is a non-economic, use-less club. <sup>2</sup>It has no headquarters because its members heads are in their hindquarters.</p> <p>(3) The fiscal year is from March 31st through April 1st.</p> <p><b>§ 2 Purpose of the Club</b></p> <p>(1) <sup>1</sup>The club is pointless but not useless. <sup>2</sup>Rather, it should put the serious handling of seriousness on a sound footing.</p> <p>(2) For this purpose, the club members can</p> <ol style="list-style-type: none"> <li>pick their noses,</li> <li>crack nuts,</li> <li>such their thumbs.</li> </ol> <p>(3) The club is selfish and stands by it.</p> <p>(4) The club has no financial means.</p> <p><b>§ 3 Club Officers</b></p> <p>(1) The club officers hold honorary positions.</p> <p>(2) <sup>1</sup>If the club had resources (see § 2 paragraph 4 sentence 1), it could afford a full-time manager. <sup>2</sup>Without the necessary funds, this is not possible.</p> <p style="text-align: center;">2</p>
<b>Preamble</b>	<b>1</b>																										
<b>A. Overview</b>	<b>2</b>																										
§ 1. Name, Legal Form, Headquarters	2																										
§ 2. Purpose of the Club	2																										
§ 3. Club Officers	2																										
<b>B. Membership</b>	<b>3</b>																										
§ 6. Becoming a Member	3																										
§ 6a. Amendment to the Previous Clause	3																										
§ 7. Termination of Membership	3																										
§ 8. General Meeting	3																										
§ 8a. Amendment to the General Meeting	4																										
<b>C. Validity</b>	<b>4</b>																										
§ 9. Effective Date	4																										
<p style="text-align: center;">§ 6 Becoming a Member</p> <p><b>B. Membership</b></p> <p><b>§ 6 Becoming a Member</b></p> <p>(1) Everyone can purchase a membership from one of the associations listed in § 4.</p> <p>(2) <sup>1</sup>To become a member, an informal application is required. <sup>2</sup>This application should be submitted in green ink on pink paper.</p> <p>(3) Membership applications cannot be rejected.</p> <p><b>§ 6a Amendment to the Previous Clause</b></p> <p><sup>1</sup>With the repeal of § 4, § 6 paragraph 1 sentence 1 has become impractical. <sup>2</sup>In its place, memberships can be inherited.</p> <p><b>§ 7 Termination of Membership</b></p> <p><sup>1</sup>Membership ends with one's life. <sup>2</sup>For non-living members, membership does not end.</p> <p><b>§ 8 General Meeting</b></p> <p>(1) A general meeting shall take place twice per year.</p> <p>(2) The interval between two general meetings shall be no more than 6 months, 1 week, and 2 days.</p> <p>(3) The invitation to the next general meeting shall be sent no earlier than 6 months from the previous general meeting.</p> <p style="text-align: center;">3</p>	<p style="text-align: center;">§ 8a Amendment to the General Meeting</p> <p><b>§ 8a Amendment to the General Meeting</b></p> <p>The general meeting may be held at the earliest 2 weeks after the invitation is received.</p> <p><b>C. Validity</b></p> <p><b>§ 9 Effective Date</b></p> <p>(1) These articles will enter into force on 11. 11. 2011 at 11:11 am.</p> <p>(2) <sup>1</sup>If any provision of these by-laws is in conflict with any other, the by-laws will be repealed on 11. 11. 2011 at 11:11 am and 11 seconds. <sup>2</sup>The club is considered to be dissolved in this case.</p> <p style="text-align: center;">4</p>																										

Figure 1: The four pages of the CfCH example of **section 8**

## 10 State of Development

Since **KOMA-Script** 3.24, the `scrjura` package has shared the version number of the classes and other important packages of **KOMA-Script**. Package `contract` on the other hand now has a new version number independent from **KOMA-Script**.

Independent from the version number you should note that so far, the interaction of the `contract` environment with the many different settings possible with other L<sup>A</sup>T<sub>E</sub>X environments, packages, or classes has not been tested. The main reason for this is that `contract` is very specialized and far beyond the author's ordinary practice. So the author mostly relies on detailed user feedback.

# Implementation

```
1 <{*package}
```

## 11 Cooperation with **hyperref**

If **hyperref** has already loaded before `contract` the package cannot work correctly. So we throw an error. Maybe it would be a good idea to make this error fatal. But currently it is only an error.

```
2 <{*init}
3 \@ifpackageloaded{hyperref}{%
4   \PackageError{contract}{Package hyperref already loaded}{%
5     If you want to use package contract with package hyperref, you have
6     to\MessageBreak
7     load package contract before package hyperref.\MessageBreak
8     To solve the problem, you just should move the loading of package
9     hyperref\MessageBreak
10    behind the loading of package contract.}%
11 }
12 </init}
```

## 12 Prerequisites

We need package **scrkbase**. We could load this also together with **tocbasic**, which is loaded later. But loading it on its own, we can require a minimum version. At least the version of 2022/11/11 would be needed, but the release was 2023/04/17 v3.39. So this is the version we request.

**Todo:** Package `contract` is no longer a **KOMA-Script** package. So it should not use the internal **KOMA-Script** package **scrkbase**. For keys of contract environments we can just load **scrbase**. For package options we should either also switch to **scrbase** or use native key-value options of L<sup>A</sup>T<sub>E</sub>X. After doing so we would be able to replace **scrkbase** at least by **scxextend**. With this package we still could use commands like `\setkomafont`. Theoretically we would also be able to still use **KOMA-Script** options, but this would still not be recommended.

```

13 <*init>
14 \RequirePackage{scrkbase}[2023/04/17]
15 </init>

And now, tocbasic.

16 <*init>
17 \RequirePackage{tocbasic}
18 </init>

```

## 13 Options

**contract** Option **contract** can be used to make the whole document to be a contract. But in this case `\if@documentcontract` you are not allowed to reuse the **contract** environment in the document. Nor is it allowed `\@documentcontractfalse` to stop or restart the contract. With older L<sup>A</sup>T<sub>E</sub>X this is done by adding `\contract` to the `\@documentcontracttrue` end of `\document`. With an up to date L<sup>A</sup>T<sub>E</sub>X we use a hook.

```

19 <*options>
20 \KOMA@ifkey{contract}{\@documentcontract}
21 \IfLTxAtLeastTF{2020/10/01}{%
22   \AddToHook{begindocument/end}{%
23     \RelaxFamilyKey[.contract.sty]{KOMA}{contract}%
24     \if@documentcontract\expandafter\contract\fi
25   }%
26 }{%
27   \g@addto@macro\document{%
28     \RelaxFamilyKey[.contract.sty]{KOMA}{contract}%
29     \if@documentcontract\expandafter\contract\fi
30   }%
31 }
32 </options>

```

**juratotoc** Allow to set the toc level of the entries. Value **true** is the same like 2, value **false** is the same like `\maxdimen`.

```

\if@juratotoc
juratoclevel (cnt.) \toclevel@cpar
33 <*options>
34 \KOMA@key{juratotoc}[true]{%
35   \KOMA@set@ifkey{juratotoc}{@tempswa}{#1}%
36   \ifx\FamilyKeyState\FamilyKeyStateProcessed
37     \if@tempswa
38       \DeclareTOCStyleEntry[level=2]{default}{cpar}%
39     \else
40       \DeclareTOCStyleEntry[level=\maxdimen]{default}{cpar}%
41     \fi
42   \else
43     \DeclareTOCStyleEntry[level=#1]{default}{cpar}%
44   \fi
45   \KOMA@kav@xreplacevalue{contract.sty}{juratotoc}{\cpartocdepth}%
46 }
47 \KOMA@kav@xadd{contract.sty}{juratotoc}{\cpartocdepth}%
48 </options>

```

**juratocnumberwidth** Indent and number width of the toc entries.

**juratocindent** **Todo:** Since we are using package **tocbasic** for the ToC entries these options (and lengths) are not needed any longer, but users should use **\DeclareTOCStyleEntry** to setup the number width and indent of the entries.

**\cpar@numberwidth** (*ilen.*)  
**\cpar@indent** (*ilen.*)

```
49 <*options>
50 \KOMA@key{juratocnumberwidth}{%
51   \DeclareTOCStyleEntry[numwidth=#1]{default}{cpar}%
52   \FamilyKeyStateProcessed
53   \KOMA@kav@replacevalue{contract.sty}{juratocnumberwidth}{#1}%
54 }
55 \KOMA@kav@add{contract.sty}{juratocnumberwidth}{2em}
56 \KOMA@key{juratocindent}{%
57   \DeclareTOCStyleEntry[indent=#1]{default}{cpar}%
58   \FamilyKeyStateProcessed
59   \KOMA@kav@replacevalue{contract.sty}{juratocindent}{#1}%
60 }
61 \KOMA@kav@add{contract.sty}{juratocindent}{1.5em}%
62 </options>
```

**juratitlepagebreak** The options sets the boolean **\if@juratitlepagebreak**.

**\if@juratitlepagebreak** If the boolean is **\iftrue** page breaks inside clause headings are allowed (which is not recommended).  
**@juratitlepagebreaktrue**  
**@juratitlepagebreakfalse**

**Todo:** Re-implementation either using a native key-value option (easy) or a new **scrbase** family.

```
63 <*options>
64 \KOMA@ifkey{juratitlepagebreak}{@juratitlepagebreak}
65 </options>
```

**parnumber** The options switches the (automatic) paragraph numbering.

**Todo:** Re-implementation either using a native key-value option (easy) or a new **scrbase** family.

```
66 <*options>
67 \newif\ifparnumber
68 \KOMA@key{parnumber}[true]{%
69   \Ifstr{#1}{auto}{%
70     \AutoPar
71     \FamilyKeyStateProcessed
72     \KOMA@kav@remove{contract.sty}{parnumber}{manual}%
73     \KOMA@kav@remove{contract.sty}{parnumber}{auto}%
74     \KOMA@kav@add{contract.sty}{parnumber}{auto}%
75   }{%
76     \Ifstr{#1}{manual}{%
77       \ManualPar
78       \FamilyKeyStateProcessed
79       \KOMA@kav@remove{contract.sty}{parnumber}{manual}%
80       \KOMA@kav@remove{contract.sty}{parnumber}{auto}%
81       \KOMA@kav@add{contract.sty}{parnumber}{manual}%
82     }%
83   }%
84 >
```

```

82   }{%
83     \KOMA@set@ifkey{parnumber}{parnumber}{#1}%
84     \KOMA@kav@replacebool{contract.sty}{parnumber}{parnumber}%
85   }%
86 }%
87 }
88 \KOMA@kav@add{contract.sty}{parnumber}{true}
89 \KOMA@kav@add{contract.sty}{parnumber}{auto}
90 </options>

```

**paragraphmark** **Todo:** Re-implementation either using a native key-value option or a new **scrbase** family **clausemark** (easy).

**markright** **markboth** **Todo:** Remove deprecated options, because they need an internal **KOMA-Script** macro.

**\Clausemark** The options are used to activate either **\markright** or **\markboth** for clauses. **\Clausemark** expects not only the title but also the number. So it differs from, e.g., **\chaptermark**, which uses the counter automatically. But maybe I will change this some time.

```

91 <*options>
92 \newcommand*{\Clausemark}[1]{%
93   \KOMA@key{clausemark}{%
94     \beginngroup
95       \KOMA@set@ncmdkey{clausemark}{@tempa}{%
96         {false}{0},{off}{0},{no}{0},%
97         {forceright}{1},%
98         {forceboth}{2},%
99         {right}{3},%
100        {both}{4}%
101      }{#1}%
102      \ifx\FamilyKeyState\FamilyKeyStateProcessed
103        \ifcase\number\@tempa
104          \endgroup
105          \let\Clausemark\@gobble
106        \or
107          \endgroup
108          \renewcommand*{\Clausemark}[1]{%
109            \markright{\csname MakeMarkcase\endcsname{##1}}}%
110        \or
111          \endgroup
112          \renewcommand*{\Clausemark}[1]{%
113            \markboth{\csname MakeMarkcase\endcsname{##1}}%
114              {\csname MakeMarkcase\endcsname{##1}}}%
115        \or
116          \endgroup
117          \renewcommand*{\Clausemark}[1]{%
118            \ifx
119              \@mkboth\@gobbletwo
120            \else
121              \markright{\csname MakeMarkcase\endcsname{##1}}}%
122            \fi}%
123        \or
124          \endgroup

```

```

125 \renewcommand*{\Clausemark}[1]{%
126 \mkboth{\csname MakeMarkcase\endcsname{##1}}%
127 {\csname MakeMarkcase\endcsname{##1}}}%
128 \else
129 \endgroup
130 \fi
131 \FamilyKeyStateProcessed
132 \else
133 \endgroup
134 \FamilyKeyStateUnknownValue
135 \fi
136 \KOMA@kav@xreplacevalue{contract.sty}{clausemark}{#1}%
137 }
138 \KOMA@kav@add{contract.sty}{clausemark}{false}
139 \@ifundefined{KOMA@DeclareDeprecatedOption}{}{%
140 \KOMA@DeclareDeprecatedOption[contract]{markright}{clausemark=forceright}%
141 \KOMA@DeclareDeprecatedOption[contract]{markboth}{clausemark=forceboth}%
142 }
143 \KOMA@key{paragraphmark}{%
144 \PackageWarningNoLine{contract}{%
145 You've used obsolete option 'paragraphmark'.\MessageBreak
146 Usage of this option is deprecated.\MessageBreak
147 You should simply replace 'paragraphmark'\MessageBreak
148 by 'clausemark'%
149 }%
150 \KOMAExecuteOptions[.contract.sty]{clausemark=#1}%
151 }
152 </options>

```

**ref** **Todo:** Re-implementation either using a native key-value option or a new **scrbase** family (easy).

**parcite**name The formatting of the references of paragraphs and sentences. There are a long a short and **sentencecite**name a numeric form.

**Todo:** Remove deprecated options, because they need an internal **KOMA-Script** macro.

**\parcite@format** Default is the long form. Corresponding values of the two helper macros are: 0 = long, 1 = **\sentencecite@format** short, 2 = numerical, -1 = nothing.

```

153 (*options)
154 \newcommand*{\parcite@format}{0}
155 \newcommand*{\sentencecite@format}{0}

```

The options can be used to change the default.

```

156 \KOMA@key{ref}{%
157 \begingroup
158 \KOMA@set@ncmdkey{ref}{@tempa}{%
159 {parlong}{1},{longpar}{1},{ParL}{1},%
160 {parshort}{2},{shortpar}{2},{ParS}{2},%
161 {parnumeric}{3},{numericpar}{3},{ParN}{3},%
162 {paroff}{4},{nopar}{4},%
163 {sentencelong}{10},{longsentence}{10},{SentenceL}{10},%

```



```

164 {sentenceshort}{20},{shortsentence}{20},{SentenceS}{20},%
165 {sentencenumeric}{30},{numericssentence}{30},{SentenceN}{30},%
166 {sentenceoff}{40},{nosentence}{40},%
167 {long}{11},%
168 {short}{22},%
169 {numeric}{33},%
170 {paragraphonly}{44},{onlyparagraph}{44},%
171 {ParagraphOnly}{44},{OnlyParagraph}{44}%
172 }{#1}%
173 \ifx\FamilyKeyState\FamilyKeyStateProcessed
174 \aftergroup\FamilyKeyStateProcessed
175 \@tempcnta=\@tempa\relax
176 \@tempcntb=\z@
177 \@whilenum \@tempcnta>9 \do{%
178 \advance\@tempcnta -10\relax
179 \advance\@tempcntb \@ne\relax
180 }%
181 \ifcase \@tempcnta
182 \or
183 \aftergroup\def\aftergroup\parcite@format
184 \aftergroup{\aftergroup0\aftergroup}%
185 \or
186 \aftergroup\def\aftergroup\parcite@format
187 \aftergroup{\aftergroup1\aftergroup}%
188 \or
189 \aftergroup\def\aftergroup\parcite@format
190 \aftergroup{\aftergroup2\aftergroup}%
191 \or
192 \aftergroup\def\aftergroup\parcite@format
193 \aftergroup{\aftergroup-\aftergroup1\aftergroup}%
194 \fi
195 \ifcase \@tempcntb
196 \or
197 \aftergroup\def\aftergroup\sentencecite@format
198 \aftergroup{\aftergroup0\aftergroup}%
199 \or
200 \aftergroup\def\aftergroup\sentencecite@format
201 \aftergroup{\aftergroup1\aftergroup}%
202 \or
203 \aftergroup\def\aftergroup\sentencecite@format
204 \aftergroup{\aftergroup2\aftergroup}%
205 \or
206 \aftergroup\def\aftergroup\sentencecite@format
207 \aftergroup{\aftergroup-\aftergroup1\aftergroup}%
208 \fi
209 \else
210 \aftergroup\FamilyKeyStateUnknownValue
211 \fi
212 \endgroup
213 \ifx\FamilyKeyState\FamilyKeyStateProcessed
214 \KOMAC@kav@removekey{contract.sty}{ref}%
215 \ifcase\parcite@format

```

```

216 \KOMA@kav@add{contract.sty}{ref}{parlong}%
217 \or
218 \KOMA@kav@add{contract.sty}{ref}{parshort}%
219 \or
220 \KOMA@kav@add{contract.sty}{ref}{parnumeric}%
221 \or
222 \KOMA@kav@add{contract.sty}{ref}{paroff}%
223 \fi
224 \ifcase\sentencecite@format
225 \KOMA@kav@add{contract.sty}{ref}{sentencelong}%
226 \or
227 \KOMA@kav@add{contract.sty}{ref}{sentenceshort}%
228 \or
229 \KOMA@kav@add{contract.sty}{ref}{sentencenumeric}%
230 \or
231 \KOMA@kav@add{contract.sty}{ref}{sentenceoff}%
232 \fi
233 \fi
234 }
235 \KOMA@kav@add{contract.sty}{ref}{parlong}%
236 \KOMA@kav@add{contract.sty}{ref}{sentencelong}%
237 \@ifundefined{KOMA@DeclareDeprecatedOption}{}{%
238 \KOMA@DeclareDeprecatedOption[contract]{parcitename}{ref=parlong}
239 \KOMA@DeclareDeprecatedOption[contract]{sentencecitename}{ref=sentencelong}
240 }
241 \</options>

Execute the options.
242 \<*postoptions>
243 \KOMAProcessOptions\relax
244 \</postoptions>

```

## 14 Contracts, Clauses, Paragraphs and Sentences

`\contract@env@type` This macro shows the currently active contract environment.

```

245 \<*body>
246 \newcommand*{\contract@env@type}{}
247 \</body>

```

`\ellipsispar` Count one or more paragraphs given by the optional argument but print `\parellipsis` instead of a real paragraph. The default is either `\dots` or `\textellipsis` if available.

```

248 \<*body>
249 \newcommand*{\ellipsispar}[1][1]{%
250 \begingroup
251 \KOMAoptions{parnumber=manual}\parellipsis\par
252 \addtocounter{par}{#1}%
253 \if@files
254 \protected@write\@auxout{}{%
255 \string\newmaxpar{\contract@env@type}%

```

```

256             {\csname the\contract@env@type
257             AbsoluteClause\endcsname}%
258             {\thepar}%
259         }%
260     \fi
261 \endgroup
262 \addtocounter{par}{-1}\refstepcounter{par}%
263 \ignorespaces
264 }
265 \newcommand*{\parellipsis}{%
266     \scr@ifundefinedorrelax{textellipsis}{\dots}{\textellipsis}%
267 }
268 \end{body}

```

**contract** It is not allowed to nest the **contract** environments, but you can end them and start them new. But this would not end the contract and start a new contract but only delay it for some other code.

```

\if@contract@skiphyperref
  contractClause (cnt.)
    \thecontractClause
\contract@Clauseformat
  \Clauseformat
    \paragraphformat
contractSubClause (cnt.)
  \thecontractSubClause
contractAbsoluteClause (cnt.)
    269 (*body)
    270 \newenvironment{contract}{%
    271     \ifx\contract@env@type\@empty
    272         \let\@doendpe\contract@doendpe
    273         \let\Clause\contract@paragraph
    274         \let\c@Clause\c@contractClause
    275         \edef\cl@Clause{\cl@Clause\cl@contractClause}%
    276         \let\SubClause\contract@subparagraph
    277         \let\c@SubClause\c@contractSubClause
    278         \edef\cl@SubClause{\cl@SubClause\cl@contractSubClause}%
    279         \let\Sentence\contract@sentence
    280         \renewcommand*{\contract@env@type}{contract}%
    281         \aliaskomafont{Clause}{contract.Clause}%
    282     \else
    283         \PackageError{contract}{nested ‘contract’ detected}{%
    284             You may not use a ‘contract’ environment inside\MessageBreak
    285             a ‘\contract@env@type’ environment or after loading\MessageBreak
    286             package ‘contract’ with option ‘\contract@env@type’!}%
    287     \fi
    288 }{}
    289 \let\if@contract@skiphyperref\iftrue
    290 \let\cl@Clause\@empty
    291 \let\cl@SubClause\@empty
    292 \newcounter{contractClause}
    293 \renewcommand*{\thecontractClause}{%
    294     {\contract@Clauseformat{\arabic{Clause}}}}
    295 \DeclareRobustCommand*{\contract@Clauseformat}[1]{\Clauseformat{#1}}
    296 \newcommand*{\Clauseformat}[1]{\S~#1}
    297 \newcounter{contractSubClause}
    298 \@addtoreset{SubClause}{Clause}
    299 \renewcommand*{\thecontractSubClause}{%
    300     {\theClause\alph{SubClause}}}
    301 \newcounter{contractAbsoluteClause}
    302 \end{body}

```

**\DeclareNewJuraEnvironment** Using `\c_atsign_strdefjuraenvironment` to define a new juristic environment. This can be done only in document preamble.

```

303 (*body)
304 \newcommand*{\DeclareNewJuraEnvironment}[1]{%
305   \@ifundefined{#1}{\expandafter\let\csname #1\expandafter\endcsname
306     \csname end#1\endcsname}{}%
307   \@ifundefined{#1}{\let\reserved@defjuraenvironment\@defjuraenvironment}{%
308     \PackageError{contract}{ignoring declaration of ‘#1’}{%
309       You’ve tried to declare jura environment ‘#1’, but
310       environment\MessageBreak
311       ‘#1’ or command
312       \expandafter\string\csname #1\endcsname\space or
313       \expandafter\string\csname end#1\endcsname\MessageBreak
314       already exists.\MessageBreak
315       Declaration will be ignored}%
316   \long\def\reserved@defjuraenvironment##1[##2]##3##4{%
317   }%
318   \kernel@ifnextchar [%]
319     {\reserved@defjuraenvironment{#1}}{\reserved@defjuraenvironment{#1} []}%
320 }
321 \@onlypreamble\DeclareNewJuraEnvironment

```

**\@defjuraenvironment** This command is used to define a new contract environment like `contract`. Several options are provided (see the user manual for details).

```

322 \DefineFamily{KOMAAarg}
323 \DefineFamilyMember{KOMAAarg}
324 \newcommand{\@defjuraenvironment}{%
325   \long\def\@defjuraenvironment#1[##2]##3##4{%
326     \let\reserved@defjuraenvironment\relax

```

The counters:

```

327   \newcounter{#1Clause}%
328   \newcounter{#1AbsoluteClause}%
329   \newcounter{#1SubClause}%
330   \FamilyCSKey[.contract.sty]{KOMAAarg}{Clause}{#1@Clause}%
331   \FamilyCSKey[.contract.sty]{KOMAAarg}{SubClause}{#1@SubClause}%
332   \FamilyCSKey[.contract.sty]{KOMAAarg}{Sentence}{#1@Sentence}%
333   \DefineFamilyKey[.contract.sty]{KOMAAarg}{ClauseNumberFormat}{%
334     \expandafter\def\csname #1@Clauseformat \endcsname####1{##1{####1}}%
335     \expandafter\edef\csname #1@Clauseformat \endcsname{%
336       \noexpand\protect\expandafter\noexpand\csname #1@Clauseformat \endcsname
337     }%
338     \FamilyKeyStateProcessed
339   }
340   \DefineFamilyKey[.contract.sty]{KOMAAarg}{ClauseFont}{%
341     \IfExistskomafont{#1.Clause}{%
342       \IfIsAliaskomafont{#1.Clause}{%
343         \expandafter\let\csname scr@fnt@instead@#1.Clause\endcsname\relax
344         \newkomafont{#1.Clause}{##1}%
345       }{\setkomafont{#1.Clause}{##1}}%
346     }{%
347       \newkomafont{#1.Clause}{##1}%

```

```

348 }%
349 }
350 \FamilyExecuteOptions[.contract.sty]{KOMAAarg}{#2}%
351 \RelaxFamilyKey[.contract.sty]{KOMAAarg}{ClauseFont}%
352 \RelaxFamilyKey[.contract.sty]{KOMAAarg}{ClauseNumberFormat}%
353 \RelaxFamilyKey[.contract.sty]{KOMAAarg}{Sentence}%
354 \RelaxFamilyKey[.contract.sty]{KOMAAarg}{SubClause}%
355 \RelaxFamilyKey[.contract.sty]{KOMAAarg}{Clause}%
356 \@ifundefined{#1@Clauseformat}{%
357   \expandafter\DeclareRobustCommand\expandafter*%
358   \csname #1@Clauseformat\endcsname[1]{\Clauseformat{##1}}%
359 }{}%

360 \expandafter\renewcommand\expandafter*\csname the#1Clause\endcsname{%
361   {\protect\@nameuse{#1@Clauseformat}}{\arabic{#1Clause}}}%

```

Environment:

```

362 \newenvironment{#1}{%
363   \par
364   \ifx\contract@env@type\@empty
365     \edef\contract@env@type{#1}%
366     \let\@doendpe\contract@doendpe
367     \expandafter\let\expandafter\c@Clause\csname c@#1Clause\endcsname
368     \edef\cl@Clause{\cl@Clause\csname cl@#1Clause\endcsname}%
369     \expandafter\let\expandafter\c@SubClause
370     \csname c@#1SubClause\endcsname
371     \edef\cl@SubClause{\cl@SubClause
372       \csname cl@#1SubClause\endcsname}%
373     \@ifundefined{#1@Clause}{%
374       \let\Clause\contract@paragraph
375     }{%
376       \expandafter\let\expandafter\Clause
377       \csname #1@Clause\endcsname
378     }%
379     \@ifundefined{#1@SubClause}{%
380       \let\SubClause\contract@subparagraph
381     }{%
382       \expandafter\let\expandafter\SubClause
383       \csname #1@SubClause\endcsname
384     }%
385     \@ifundefined{#1@Sentence}{%
386       \let\Sentence\contract@sentence
387     }{%
388       \expandafter\let\expandafter\Sentence\csname #1@Sentence\endcsname
389     }%
390     \@ifundefined{\contract@env@type @everypar}{%
391       \expandafter\let
392       \csname \contract@env@type @everypar\endcsname
393       \contract@everypar
394     }{}%

```

Font alias for Clause. If neither a font not an alias is defined for the new environment `contract.Clause` is used.

```

395     \IfExistskomafont{#1.Clause}{%
396     \IfIsAliaskomafont{#1.Clause}{%
397     \aliaskomafont{Clause}{\csname scr@fnt@instead@#1.Clause\endcsname}%
398     }{%
399     \aliaskomafont{Clause}{#1.Clause}%
400     }%
401     }{%
402     \aliaskomafont{Clause}{contract.Clause}%
403     }%
404     #3%
405     \else
406     \PackageError{contract}{nested contract environments detected}{%
407     You must not use a ‘#1’ environment inside\MessageBreak
408     a ‘\contract@env@type’ environment or after loading\MessageBreak
409     package ‘contract’ with option ‘\contract@env@type’!}%
410     \fi
411     }{%
412     #4%
413     \par
414     }%
415 }
416 \end{body}

```

`\contract@paragraph` This is the `\Clause` used by contracts. A contract consists (usually) of several clauses. Each clause has optional elements managed by  $\langle key \rangle = \langle value \rangle$  pairs handled by `scrkbase` and last but not least by `keyval`.

`title` Title, running head and toc entry of the clause. The title is the default for running head and head toc entry. But you can also use an empty value for each of them or use the `no...` options `nohead` to switch them off.

```

entry 417 (*body)
noentry 418 \define@key{contract}{title}{%
tocentry 419 \def\contract@title{#1}%
notocentry 420 \ifx\contract@entry\relax\def\contract@entry{\contract@title}\fi
421 \ifx\contract@head\relax\def\contract@head{\contract@title}\fi
422 }
423 \define@key{contract}{entry}{%
424 \PackageWarning{contract}{deprecated option ‘entry’.\MessageBreak
425 You should use option ‘tocentry’ instead of\MessageBreak
426 option ‘entry’%
427 }%
428 \def\contract@entry{#1}}
429 \define@key{contract}{tocentry}{\def\contract@entry{#1}}
430 \define@key{contract}{noentry}[]{%
431 \PackageWarning{contract}{deprecated option ‘noentry’.\MessageBreak
432 You should use option ‘notocentry’ instead of\MessageBreak
433 option ‘noentry’%
434 }%
435 \let\contract@entry\relax}
436 \define@key{contract}{notocentry}[]{\let\contract@entry\relax}
437 \define@key{contract}{head}{\def\contract@head{#1}}
438 \define@key{contract}{nohead}[]{\let\contract@head\relax}

```

**number** The number can be changed manually. But clauses without numbers are not allowed. So if you use an empty value, the number is automatically set.

```
439 \define@key{contract}{number}{\def\contract@number{#1}}
```

**\contract@preskip** The options are used to specify the distance before and after the clause. The preset value of these options are the global settings done by `\setkeys{contract}{...}`.

```
preskip 440 \newcommand*{\contract@preskip}{2\baselineskip}
postskip 441 \newcommand*{\contract@postskip}{\baselineskip}
442 \define@key{contract}{preskip}{\def\contract@preskip{#1}}
443 \define@key{contract}{postskip}{\def\contract@postskip{#1}}
```

**dummy** The option switches the boolean `\ifcontract@dummy`.

**\ifcontract@dummy** If the boolean is `\iftrue` the clause will not be printed. But note: you cannot use this to remove the paragraphs or sentences of the clause. But you can use this option to generate holes in the numbering without manually manipulating the counters.

```
444 \newif\ifcontract@dummy
445 \define@key{contract}{dummy}[true]{\csname contract@dummy#1\endcsname}
```

**contract.Clause (font)** Correctly this macro should be named `\contract@paragraph@format`. But it is already used by some users for ugly tricks. So I will not rename it to avoid problems for existing documents. Additionally it would be better to use a new macro per environment. However the same reason not to change this.

```
446 \newkomafont{contract.Clause}{\sffamily\bfseries\large}
447 \newcommand*{\contract@paragraph@font}{\usekomafont{Clause}}%
448 \@hangfrom}
```

**@AbsClause (cnt.)**

```
\theH@AbsClause 449 % Here we have some not good tested code for \pkg{hyperref}.
\theHClause 450 \newcounter{@AbsClause}
\theHSubClause 451 \def\theH@AbsClause{P-\arabic{@AbsClause}}
452 \def\theHClause{\theH@AbsClause}
453 \def\theHSubClause{\theH@AbsClause}
```

For the headings we use manual paragraph numbering, because we don't want any paragraph numbering inside the heading. After initializing the options they are processed.

```
454 \NewDocumentCommand\contract@paragraph {o} {%
455   \stepcounter{\contract@env@type AbsoluteClause}%
456   \ManualPar\parnumbertrue
457   \let\contract@title\relax
458   \let\contract@entry\relax
459   \let\contract@head\relax
460   \let\contract@number\relax
461   \contract@dummyfalse
462   \IfValueT{#1}{\setkeys{contract}{#1}}%
```

Unless this is a dummy clause, the headings will be initialized and vertical skips will be done.

```
463   \ifcontract@dummy\else
464     \par
465     \@afterindentfalse
```

```

466 \addvspace{\contract@preskip}%
467 \fi

```

If there isn't a manual number, we use the next number. If there is a manual number, this number is printed and we take care that labels and [hyperref](#) also use the manual number.

```

468 \ifx\contract@number\relax
469 \let\p@Clause\@empty
470 \expandafter\let\expandafter\theClause
471 \csname the\contract@env@type Clause\endcsname
472 \refstepcounter{Clause}%
473 \else
474 \begingroup
475 \let\@elt\@stpelt
476 \cl@Clause
477 \endgroup

478 \protected@edef\theClause{%
479 \protect\@nameuse{\contract@env@type @Clauseformat}{\contract@number}%
480 }%
481 \protected@edef\@currentlabel{\theClause}%
482 \def\@currentcounter{Clause}%
483 \fi
484 \stepcounter{@AbsClause}%
485 \begingroup\expandafter\expandafter\expandafter\endgroup
486 \expandafter\ifx\csname if@skiphyperref\endcsname\relax
487 \else
488 \expandafter\let\csname if@contract@skiphyperref\endcsname
489 \csname if@skiphyperref\endcsname
490 \fi
491 \if@contract@skiphyperref\else
492 \hyper@refstepcounter{@AbsClause}%
493 \ifx\@currentHref\@currentHref\relax
494 \fi

```

For simplification we use the code of clauses for sub-clauses.

```

495 \let\theSubClause\theClause

```

Unless for dummy clauses, the heading is printed, the toc entry is done and also the running head.

```

496 \ifcontract@dummy\else
497 \begingroup
498 \if@juratitlepagebreak\else\interlinepenalty\@M\fi
499 \contract@paragraph@font{\theClause
500 \ifx\contract@title\relax\else\enskip\fi}%
501 \contract@title
502 \ifx\contract@entry\relax\else
503 \expandafter\addxcontentsline\expandafter{\ext@toc}%
504 {cpar}[\theClause]\contract@entry
505 \addxcontentsline{cpar}{cpar}[\theClause]\contract@entry
506 \fi
507 \ifx\contract@head\relax\else
508 \expandafter\Clausemark\expandafter{%
509 \expandafter\theSubClause\expandafter\enskip\contract@head}%

```



```

510     \fi
511     \par
512     \endgroup\nobreak\vskip\contract@postskip

```

Last but not least paragraph numbering is initialized.

```

513     \contract@afterheading
514     \fi
515 }
516 </body>

```

`\contract@subparagraph` This is almost the same like `\contract@paragraph`.

```

517 (*body)
518 \NewDocumentCommand \contract@subparagraph {o}{%
519   \stepcounter{\contract@env@type AbsoluteClause}%
520   \ManualPar\parnumbertrue
521   \let\contract@title\relax
522   \let\contract@entry\relax
523   \let\contract@head\relax
524   \let\contract@number\relax
525   \contract@dummysfalse
526   \IfValueT{#1}{\setkeys{contract}{#1}}%
527   \ifcontract@dummysfalse
528     \par
529     \@afterindentfalse
530     \vskip\contract@preskip
531     \fi
532     \ifx\contract@number\relax
533       \let\p@SubClause\@empty
534       \let\theSubClause\thecontractSubClause
535       \refstepcounter{SubClause}%
536     \else
537       \begingroup
538         \let\@elt\@stpelt
539         \cl@SubClause
540       \endgroup
541       \protected@edef\theSubClause{\theClause\contract@number}%
542       \protected@edef\@currentlabel{\theSubClause}%
543       \def\@currentcounter{SubClause}%
544     \fi
545     \stepcounter{@AbsClause}%
546     \begingroup\expandafter\expandafter\expandafter\endgroup
547     \expandafter\ifx\csname if@skiphyperref\endcsname\relax
548     \else
549       \expandafter\let\csname if@contract@skiphyperref\endcsname
550       \csname if@skiphyperref\endcsname
551     \fi
552     \if@contract@skiphyperref\else
553       \hyper@refstepcounter{@AbsClause}%
554 (+trace) \typeout{absolute Number: \the@AbsClause^^JLabel: '\@currentHref'}%
555     \fi
556     \ifcontract@dummysfalse
557       \begingroup

```

```

558 \if@juratitlepagebreak\else\interlinepenalty\@M\fi
559 \contract@paragraph@font{\theSubClause
560 \ifx\contract@title\relax\else\enskip\fi}%
561 \contract@title
562 \ifx\contract@entry\relax\else
563 \expandafter\addxcontentsline\expandafter{\ext@toc}%
564 {cpar}[\theSubClause]\contract@entry
565 \addxcontentsline{cpar}{cpar}[\theSubClause]\contract@entry
566 \fi
567 \ifx\contract@head\relax\else
568 \expandafter\Clausemark\expandafter{%
569 \expandafter\theSubClause\expandafter\enskip\contract@head}%
570 \fi
571 \par
572 \endgroup
573 \nobreak\vskip\contract@postskip
574 \contract@afterheading
575 \fi
576 }
577 </body>

```

`\AutoPar` Switching between automatic or manual paragraph numbers for all contract environments.

```

\ManualPar 578 <*body>
579 \newcommand*{\AutoPar}{%
580 \expandafter\let\expandafter\contract@used@everypar
581 \csname \contract@env@type @everypar\endcsname
582 }
583 \newcommand*{\ManualPar}{%
584 \let\contract@used@everypar\relax
585 }
586 </body>

```

`\contract@afterheading` Similar to `\afterheading` but with automatic paragraph numbers.

**ToDo:** Test if this can be done using L<sup>A</sup>T<sub>E</sub>X hooks, depending on the L<sup>A</sup>T<sub>E</sub>X release.

```

587 <*body>
588 \CheckCommand*{\@afterheading}{%
589 \@nobreaktrue
590 \everypar{%
591 \if@nobreak
592 \@nobreakfalse
593 \clubpenalty \@M
594 \if@afterindent \else
595 {\setbox\z@\lastbox}%
596 \fi
597 \else
598 \clubpenalty \@clubpenalty
599 \everypar{}%
600 \fi}%
601 }
602 \newcommand*{\contract@afterheading}{%

```

```

603 \@nbreaktrue
604 \everypar{%
605   \if@nbreak
606     \@nbreakfalse
607     \clubpenalty \@M
608     \if@afterindent \else
609       {\setbox\z@\lastbox}%
610     \fi
611   \else
612     \clubpenalty \@clubpenalty
613     \everypar{%
614       \contract@used@everypar
615     }%
616   \fi
617   \contract@used@everypar
618 }%
619 \AutoPar
620 }

```

`\contract@used@everypar` The macro to be used at the very beginning of every paragraph to add the number. To be used only inside `contract` environments, so empty outside.

```
621 \newcommand*{\contract@used@everypar}{}

```

`\@doendpe` L<sup>A</sup>T<sub>E</sub>X used this macro, to reset all paragraph actions at the end of environments. To avoid unwanted switching-off of the paragraph number it will be reinitialized.

`\contract@doendpe` From L<sup>A</sup>T<sub>E</sub>X 2015/01/01 a different definition of `\c_atsign_strdoendpe` is used. So we also

`\IncludeInRelease` have to use different versions depending on the release. We do so with some tricks. Maybe

`\@gobble@IncludeInRelease` this should be replaced by usage of `\IfLTxAtLeastTF` from already loaded `scrbase`.

`\EndIncludeInRelease` **Todo:** Using `\everypar` is evil and should be replaced by using a generic paragraph hook.

```

622 \providecommand*{\IncludeInRelease}[3]{%
623   \PackageInfo{contract}{temporary definition of \string\IncludeInRelease}%
624   \Ifstr{#1}{0000/00/00}{%
625     \let\IncludeInRelease\@undefined
626     \def\EndIncludeInRelease{\let\EndIncludeInRelease\@undefined}%
627   }{%
628     \let\EndIncludeInRelease\relax
629     \long\def\@gobble@IncludeInRelease##1\EndIncludeInRelease{%
630       \let\@gobble@IncludeInRelease\@undefined
631     }%
632     \expandafter\@gobble@IncludeInRelease
633   }%
634 }
635 \IncludeInRelease{2015/01/01}{\@doendpe}{\clubpenalty fix}
636 \CheckCommand*\@doendpe{\@endpetrue
637   \def\par{\@restorepar
638     \clubpenalty\@clubpenalty
639     \everypar{}\par\@endpefalse}\everypar
640     {\setbox\z@\lastbox}%
641     \everypar{}\@endpefalse}}
642 \newcommand*{\contract@doendpe}{%

```

```

643 \endpetrue
644 \def\par{%
645 \restorepar
646 \clubpenalty\@clubpenalty
647 \everypar{%
648 \csname contract@used@everypar\endcsname
649 }%
650 \par\@endpefalse
651 }%
652 \everypar{%
653 {\setbox\z@\lastbox}\everypar{%
654 \csname contract@used@everypar\endcsname
655 }%
656 \@endpefalse
657 }%
658 }
659 \EndIncludeInRelease
660 \IncludeInRelease{0000/00/00}{\@doendpe}{clubpenalty fix}
661 \CheckCommand*\@doendpe{\@endpetrue
662 \def\par{\@restorepar\everypar{}\par\@endpefalse}\everypar
663 {\setbox\z@\lastbox}\everypar{}\@endpefalse}}
664 \newcommand*{\contract@doendpe}{%
665 \@endpetrue
666 \def\par{%
667 \restorepar\everypar{%
668 \csname contract@used@everypar\endcsname
669 }%
670 \par\@endpefalse
671 }%
672 \everypar{%
673 {\setbox\z@\lastbox}\everypar{%
674 \csname contract@used@everypar\endcsname
675 }%
676 \@endpefalse
677 }%
678 }
679 \EndIncludeInRelease
680 \</body>

```

## 15 Entry to Table of Contents

\l@cpar Toc entry of contract clauses. This is done using [tocbasic](#). The definition has to be part of the initialization of the package, otherwise package options wouldn't be able to change the setting.

```

681 \<init>
682 \DeclareTOCStyleEntry[%
683 indent=1.5em,
684 numwidth=2em,
685 level=\maxdimen
686 ]{default}{cpar}

```

687 `\init`

## 16 Numbering of Paragraphs and Sentences

`\contract@separator` Used to make it possible to remove white spaces at the beginning or end.

```
688 \begin{body}
689 \DeclareRobustCommand*\contract@separator}[1]{#1}
690 \end{body}
```

`\contract@usetype` By default it is robust but does only call `\contract@@usetype` with the only argument.

`\contract@@usetype` This second command is not robust and can easily be redefined. But by default it also does nothing but eating the argument.

```
691 \begin{body}
692 \DeclareRobustCommand*\contract@usetype}[1]{\contract@@usetype{#1}}
693 \newcommand*\contract@@usetype}[1]{}
694 \end{body}
```

`\contract@everypar` The `\contract@everypar` used by contracts.

`\ifparnumber` The boolean defines if paragraph numbers have to be used. If they are deactivated also

`\parnumbertrue` manual paragraph numbers are deactivated and the paragraphs are not counted. Otherwise

`\parnumberfalse` the paragraphs are numbered using `\thepar`. It is important to reset the paragraph counter

`par (cnt.)` with every clause and sub-clause. And for labels the parent object the clause has to be used.

`\thepar` 695 `\begin{body}`

`\theHpar` 696 `\newcounter{par}`

`\parformat` 697 `\renewcommand*\thepar{\arabic{par}}`

`\parformatseparation` 698 `\def\theHpar{\theH@AbsClause-\Roman{par}}`

`\p@par` 699 `\newcommand*\parformat{(\thepar)}`

`\withoutparnumber` 700 `\newcommand*\parformatseparation{\nobreakspace}`

701 `\newkomafont{parnumber}{}`

702 `\renewcommand*\p@par{\contract@usetype{\contract@env@type}\theSubClause\contract@separator}`

703 `\@addtoreset{par}{Clause}`

704 `\@addtoreset{par}{SubClause}`

705 `\newcommand*\withoutparnumber{}`

706 `\end{body}`

707 `\begin{body}`

708 `\newcommand*\contract@everypar}{%`

709 `\ifparnumber`

710 `\ifx\contract@special@par\relax`

711 `\ifx\contract@special@reset@par\relax\else`

712 `\global\let\thepar\contract@special@reset@par`

713 `\global\let\contract@special@reset@par\relax`

714 `\fi`

715 `\refstepcounter{par}%`

716 `\refstepcounter{sentence}%`

717 `\else`

718 `\ifx\contract@special@reset@par\relax`

719 `\global\let\contract@special@reset@par\thepar`

720 `\fi`

```

721 \global\let\thepar\contract@special@par
722 \global\let\contract@special@par\relax
723 \setcounter{sentence}{0}\refstepcounter{sentence}%
724 \fi
725 \begingroup
726 \if@filesw
727 \protected@write\@auxout{%
728 \expandafter\let\csname \contract@env@type @Clauseformat\endcsname
729 \@firstofone
730 }{%
731 \string\newmaxpar{\contract@env@type}%
732 {\csname the\contract@env@type
733 AbsoluteClause\endcsname}%
734 {\thepar}%
735 }%
736 \fi
737 \getmaxpar\@tempa{\contract@env@type}%
738 {\csname the\contract@env@type AbsoluteClause\endcsname}%
739 <+trace> \typeout{Stored max is \@tempa}%
740 \def\reserved@a##1\@nnil{\def\@tempa{##1}}%
741 \afterassignment\reserved@a\@tempcmta=0\@tempa\relax\@nnil
742 \ifnum \@tempcmta>\@ne
743 {\usekomafont{parnumber}{\parformat\parformatseparation}}%
744 \else
745 \def\reserved@a{\relax}%
746 \ifx\@tempa\reserved@a
747 \withoutparnumber
748 \else
749 {\usekomafont{parnumber}{\parformat\parformatseparation}}%
750 \fi
751 \fi
752 \endgroup
753 \else
754 \begingroup\withoutparnumber\endgroup
755 \setcounter{sentence}{-1}\refstepcounter{sentence}%
756 \fi
757 }
758 </body>

```

\thisparnumber You can use this for manual paragraph numbering. But the number has to be fully expand-  
\contract@special@par able!  
\contract@special@reset@par

```

759 <*body>
760 \newcommand*{\thisparnumber}[1]{%
761 \def\contract@special@par{#1}%
762 }
763 \newcommand*{\contract@special@par}{}
764 \let\contract@special@par\relax
765 \newcommand*{\contract@special@reset@par}{}
766 \let\contract@special@reset@par\relax
767 </body>

```

## 17 Referencing

**\refL** Similar to **\ref** but always the long form.

```
\ref@L 768 <*body>
769 \newcommand*{\refL}{\kernel@ifstar {\ref@L*}{\ref@L{}}}
770 \newcommand*{\ref@L}[2]{%
771   \begingroup
772   \def\parcite@format{0}%
773   \let\sentencecite@format\parcite@format
774   \ref#1{#2}%
775   \endgroup
776 }
```

**\refS** Similar to **\ref** but always the short form.

```
\ref@S 777 \newcommand*{\refS}{\kernel@ifstar {\ref@S*}{\ref@S{}}}
778 \newcommand*{\ref@S}[2]{%
779   \begingroup
780   \def\parcite@format{1}%
781   \let\sentencecite@format\parcite@format
782   \ref#1{#2}%
783   \endgroup
784 }
```

**\refN** Similar to **\ref** but always the numerical form.

```
\ref@N 785 \newcommand*{\refN}{\kernel@ifstar {\ref@N*}{\ref@N{}}}
786 \newcommand*{\ref@N}[2]{%
787   \begingroup
788   \def\parcite@format{2}%
789   \let\sentencecite@format\parcite@format
790   \ref#1{#2}%
791   \endgroup
792 }
```

**\refClause** Reference only the clause of a clause, paragraph or sentence. For better compatibility with **\ref@Clause** **hyperref** there is also a star version if **hyperref** is used. Without **hyperref** the star version is nonsense.

```
793 \newcommand*{\refClause}{%
794   \kernel@ifstar {\ref@Clause*}{\ref@Clause{}}
795 }
796 \newcommand*{\ref@Clause}[2]{%
797   \expandafter\ifx\csname r@#2\endcsname\relax
798     \ref#1{#2}%
799   \else
800     \begingroup
```

Copy all parts of the reference but the first one to `\c_atsign_strtempb`.

```
801   \expandafter\expandafter\expandafter\expandafter
802   \expandafter\expandafter\expandafter\def
803   \expandafter\expandafter\expandafter\expandafter
```

```

804 \expandafter\expandafter\expandafter\@tempb
805 \expandafter\expandafter\expandafter\expandafter
806 \expandafter\expandafter\expandafter{%
807 \expandafter\expandafter\expandafter\@gobble\csname r@#2\endcsname}%
Copy the first part of the reference to \c_atsign_strtempa.
808 \def\@tempc##1##2\@nil{##1}%
809 \let\contract@separator\@gobble
810 \protected@edef\@tempa{\expandafter\expandafter\expandafter\@tempc
811 \csname r@#2\endcsname\noexpand\@nil}%
Copy the first part of \c_atsign_strtempa to \c_atsign_strtempb.
812 \protected@edef\@tempa{\expandafter\expandafter\expandafter\@tempc
813 \@tempa\@nil}%
814 \let\@@protect\protect
815 \let\protect\noexpand
816 \expandafter\edef\csname r@#2\endcsname{\@tempa\@tempb}%
817 \let\protect\@@protect
818 \ref#1{#2}%
819 \endgroup
820 \fi
821 }

```

**\refClauseN** Reference only the clause number of a clause, a paragraph or a sentence. For improved **\ref@ClauseN** compatibility with **hyperref** there is also a star version if **hyperref** is used. Without **hyperref** the star version is nonsense.

```

822 \newcommand*{\refClauseN}{%
823 \kernel@ifstar {\ref@ClauseN*}{\ref@ClauseN{}}
824 }
825 \newcommand*{\ref@ClauseN}[2]{%
826 \begingroup
827 \let\Clauseformat\relax
828 \ref@Clause{#1}{#2}%
829 \endgroup
830 }

```

**\refPar** References only the paragraph of a paragraph or sentence. For improved compatibility with **\ref@Par** **hyperref** there is also a star version if **hyperref** is used. Without **hyperref** the star version is nonsense.

```

831 \newcommand*{\refPar}{%
832 \kernel@ifstar {\ref@Par*}{\ref@Par{}}
833 }
834 \newcommand*{\ref@Par}[2]{%
835 \expandafter\ifx\csname r@#2\endcsname\relax
836 \ref#1{#2}%
837 \else
838 \begingroup

```

Copy all parts of the reference but the first one to \c\_atsign\_strtempb.

```

839 \expandafter\expandafter\expandafter\expandafter
840 \expandafter\expandafter\expandafter\def

```



```

841 \expandafter\expandafter\expandafter\expandafter
842 \expandafter\expandafter\expandafter\@tempb
843 \expandafter\expandafter\expandafter\expandafter
844 \expandafter\expandafter\expandafter{%
845 \expandafter\expandafter\expandafter\@gobble\csname r@#2\endcsname}%

```

Copy the first part of the reference to `\c_atsign_strtempa`.

```

846 \def\@tempc##1##2\@nil{##1}%
847 \let\contract@separator\@gobble
848 \protected@edef\@tempa{\expandafter\expandafter\expandafter\@tempc
849 \csname r@#2\endcsname\noexpand\@nil}%

```

Copy the second part of `\c_atsign_strtempa` to `\c_atsign_strtempa` ablegen.

```

850 \def\@tempc##1##2##3\@nil{##2}%
851 \protected@edef\@tempa{\expandafter\expandafter\expandafter\@tempc
852 \@tempa{%
853 \protect\G@refundefinedtrue
854 \nfss@text{\reset@font\bfseries ??}%
855 \@latex@warning{Reference ‘#2’ on page \thepage \space
856 with undefined par number}%
857 }\noexpand\@nil}%
858 \let\@@protect\protect
859 \let\protect\noexpand
860 \expandafter\edef\csname r@#2\endcsname{{\@tempa}\@tempb}%
861 \let\protect\@@protect
862 \ref#1{#2}%
863 \endgroup
864 \fi
865 }

```

**\refParL** The same but long.

```

\ref@ParX 866 \newcommand*\refParL{%
867 \kernel@ifstar {\ref@ParX0*}{\ref@ParX0{}}
868 }
869 \newcommand*\ref@ParX}[3]{%
870 \begingroup
871 \def\parcite@format{#1}%
872 \let\sentencecite@format\parcite@format
873 \ref@Par{#2}{#3}%
874 \endgroup
875 }

```

**\refParS** The same but short.

```

876 \newcommand*\refParS{%
877 \kernel@ifstar {\ref@ParX1*}{\ref@ParX1{}}
878 }

```

**\refParN** The same but numerical.

```

\ref@ParN 879 \newcommand*\refParN{%
\ref@@ParN 880 \kernel@ifstar {\ref@ParN2*}{\ref@ParN2{}}

```

```

881 }
882 \newcommand*{\ref@ParN}[2]{%
883   \kernel@ifnextchar [%]
884     {\ref@ParN{#1}{#2}}%
885     {\ref@ParX{#1}{#2}}%
886 }
887 \newcommand*{\ref@ParN}{}
888 \def\ref@ParN#1#2[#3]#4{%
889   \begingroup
890     \renewcommand*{\parnumericformat}[1]{%
891       \csname @#3\endcsname{\number ##1\relax}%
892     }%
893     \ref@ParX{#1}{#2}{#4}%
894   \endgroup
895 }

```

**\refSentence** Reference only the sentence of a sentence. For improved compatibility with **hyperref** there is also a star version if **hyperref** is used. Without **hyperref** the star version is nonsense.

```

896 \newcommand*{\refSentence}{%
897   \kernel@ifstar {\ref@Sentence*}{\ref@Sentence{}}
898 }
899 \newcommand*{\ref@Sentence}[2]{%
900   \expandafter\ifx\csname r@#2\endcsname\relax
901     \ref#1{#2}%
902   \else
903     \begingroup

```

Copy all parts of the reference to \c\_atsign\_strtempb.

```

904     \expandafter\expandafter\expandafter\expandafter
905     \expandafter\expandafter\expandafter\def
906     \expandafter\expandafter\expandafter\expandafter
907     \expandafter\expandafter\expandafter\@tempb
908     \expandafter\expandafter\expandafter\expandafter
909     \expandafter\expandafter\expandafter{%
910       \expandafter\expandafter\expandafter\@gobble\csname r@#2\endcsname}%

```

Copy the first part of the reference to \c\_atsign\_strtempa.

```

911     \def\@tempc##1##2\@nil{##1}%
912     \let\contract@separator\@gobble
913     \protected@edef\@tempa{\expandafter\expandafter\expandafter\@tempc
914       \csname r@#2\endcsname\noexpand\@nil}%

```

Copy the third part of \c\_atsign\_strtempa to \c\_atsign\_strtempa.

```

915     \def\@tempc##1##2##3##4\@nil{##3}%
916     \protected@edef\@tempa{\expandafter\expandafter\expandafter\@tempc
917       \@tempa}{%
918       \protect\G@refundefinedtrue
919       \nfss@text{\reset@font\bfseries ??}%
920       \@latex@warning{Reference ‘#2’ on page \thepage \space
921         with undefined sentence number}%
922     }\noexpand\@nil}%
923     \let\@@protect\protect

```

```

924     \let\protect\noexpand
925     \expandafter\edef\csname r@#2\endcsname{\@tempa}\@tempb}%
926     \let\protect\@protect
927     \ref#1{#2}%
928   \endgroup
929 \fi
930 }

```

**\refSentenceL** The same but long.

```

\ref@SentenceX 931 \newcommand*\refSentenceL{%
932   \kernel@ifstar {\ref@SentenceX0*}{\ref@SentenceX0{}}
933 }
934 \newcommand*\ref@SentenceX[3]{%
935   \begingroup
936   \def\parcite@format{#1}%
937   \let\sentencecite@format\parcite@format
938   \ref@Sentence{#2}{#3}%
939   \endgroup
940 }

```

**\refSentenceS** The same but short.

```

941 \newcommand*\refSentenceS{%
942   \kernel@ifstar {\ref@SentenceX1*}{\ref@SentenceX1{}}
943 }

```

**\refSentenceN** The same but numeric.

```

944 \newcommand*\refSentenceN{%
945   \kernel@ifstar {\ref@SentenceX2*}{\ref@SentenceX2{}}
946 }
947 </body>

```

**\contract@sentence** Numbering of sentences.

**sentence** (*cnt.*) The counter is used for numbering the sentences. It is important to add the paragraph as **\thesentence** parent object to labels. The original method to make it possible to use **\thesentence** as **\theHsentence** an argument of **\p@sentence** does not work any longer using L<sup>A</sup>T<sub>E</sub>X 2019-10-01 or newer. It **\p@sentence** would result in an error message. So the code has to be adapted to the new definition of **\refstepcounter** in L<sup>A</sup>T<sub>E</sub>X 2019-10-01. From this version it uses **\labelformat**. Don't ask me, what I think about the fact, that every new versions of L<sup>A</sup>T<sub>E</sub>X can break existing packages and package authors have to find out such incompatibilities on their own.

```

948 <body>
949 \newcounter{sentence}[par]
950 \renewcommand*\thesentence{\arabic{sentence}}
951 \def\theHsentence{\theHpar-\arabic{sentence}}
952 \scr@ifundefinedorrelax{labelformat}{%
953   \renewcommand*\p@sentence{\expandafter\p@@sentence}
954   \newcommand*\p@@sentence[1]{\p@par{\par@cite{\thepar}}%
955     \contract@separator{\nobreakspace}}{\sentence@cite{#1}}}%
956 }{%

```

```

957 \labelformat{sentence}{\p@par{\par@cite{\thepar}}}%
958 \contract@separator{\nobreakspace}{\sentence@cite{#1}}}%
959 }
960 \newcommand*{\contract@sentence}{%

```

For the numbering it is important not to increase the paragraph number at the very beginning, because the paragraph already does so. To make this work, the paragraph has to start before we print the number. But immediately after a `minipage`, a list or a `\parbox` we should behave as not being at the beginning of a paragraph.

```

961 \ifvmode
962 \if@endpe
963 \refstepcounter{sentence}%
964 \else
965 \leavevmode
966 \fi
967 \else
968 \refstepcounter{sentence}%
969 \fi
970 {\usekomafont{sentence}{\sentenceformat}}%
971 \nobreak\hskip\z@
972 }

```

`sentence` (*font*) Formatting an font can be changed using font element `sentence` and command `\sentenceformat` `\sentenceformat`. The last has the preset `\textsuperscript`.

```

973 \newkomafont{sentence}{}
974 \newcommand*{\sentenceformat}{\textsuperscript{\thesentence}}
975 \end{body}

```

`\par@cite` Reference style for paragraphs.

```

\parciteformat 976 (*body)
977 \DeclareRobustCommand*{\par@cite}[1]{\parciteformat{#1}}
978 \newcommand*{\parciteformat}[1]{%
979 \ifcase \parcite@format
980 \expandafter\parlongformat
981 \or
982 \expandafter\parshortformat
983 \or
984 \expandafter\parnumericformat
985 \else
986 \unskip\expandafter\@gobble
987 \fi
988 {#1}%
989 }

```

`\sentence@cite` Reference style for sentences. Preset is `\c_atsign_strarabic`.

```

\sentenceciteformat 990 \DeclareRobustCommand*{\sentence@cite}[1]{\sentenceciteformat{#1}}
991 \newcommand*{\sentenceciteformat}[1]{%
992 \ifcase \sentencecite@format
993 \expandafter\sentencelongformat
994 \or
995 \expandafter\sentenceshortformat
996 \or

```

```

997     \expandafter\sentencenumericformat
998     \else
999     \unskip\expandafter\@gobble
1000     \fi
1001     {#1}%
1002 }

```

```

\parlongformat The six formattings.
\parshortformat 1003 \newcommand*{\parlongformat}[1]{\parname~#1}
\parnumericformat 1004 \newcommand*{\parshortformat}[1]{\parshortname~#1}
\sentencelongformat 1005 \newcommand*{\parnumericformat}[1]{\@Roman{\number #1\relax}}
\sentenceshortformat 1006 \newcommand*{\sentencelongformat}[1]{\sentencename~#1}
\sentencenumericformat 1007 \newcommand*{\sentenceshortformat}[1]{\sentenceshortname~#1}
1008 \newcommand*{\sentencenumericformat}[1]{\@arabic{\number #1\relax}.}
1009 \</body>

```

## 18 Language Dependent Names

```

\parname The names of paragraphs and sentences and their short versions. The English names are
\parshortname donated by “m.eik”.
\sentencename 1010 \<*body>
\sentenceshortname 1011 \newcommand*{\parname}{Paragraph}
\contract@lang@error 1012 \AtBeginDocument{%
1013     \providecaptionname{german,ngerman,austrian,naustrian}\parname{Absatz}%
1014     \providecaptionname{german,ngerman,austrian,naustrian}\parshortname{Abs.}%
1015     \providecaptionname{german,ngerman,austrian,naustrian}\sentencename{Satz}%
1016     \providecaptionname{german,ngerman,austrian,naustrian}\sentenceshortname{S.}%
1017     \providecaptionname{english,american,british,canadian,%
1018         USenglish,UKenglish,usenglish,ukenglish}\parname{paragraph}%
1019     \providecaptionname{english,american,british,canadian,%
1020         USenglish,UKenglish,usenglish,ukenglish}\parshortname{par.}%
1021     \providecaptionname{english,american,british,canadian,%
1022         USenglish,UKenglish,usenglish,ukenglish}\sentencename{sentence}%
1023     \providecaptionname{english,american,british,canadian,%
1024         USenglish,UKenglish,usenglish,ukenglish}\sentenceshortname{sent.}%
1025 }
1026 \providecommand*{\parname}{\contract@lang@error{\parname}}
1027 \providecommand*{\parshortname}{\contract@lang@error{\parshortname}}
1028 \providecommand*{\sentencename}{\contract@lang@error{\sentencename}}
1029 \providecommand*{\sentenceshortname}{\contract@lang@error{\sentenceshortname}}
1030 \newcommand*{\contract@lang@error}[1]{%
1031     \PackageError{contract}{%
1032         current language not supported%
1033     }{%
1034         Currently contract only supports languages ‘german’, ‘ngerman’,
1035         ‘austrian’,\MessageBreak
1036         ‘naustrian’, ‘english’, ‘american’, ‘british’, ‘canadian’,
1037         ‘USenglish’,\MessageBreak
1038         ‘UKenglish’, ‘usenglish’, and ‘ukenglish’.\MessageBreak
1039         It seems, that you are using another language (maybe ‘\language’) or

```

```

1040     that\MessageBreak
1041     your language selection isn't compatible to package 'babel'.\MessageBreak
1042     Because of this you have to define '\string#1' by yourself!\MessageBreak
1043     It would be nice if you'll send your definitions to the author.%
1044 }%
1045 \textbf{??}%
1046 }
1047 </body>

```

## 19 Using Values from last $\text{\LaTeX}$ Run

`\newmaxpar` Two helper macros, to save a counter in a aux-file and get the value back or another value  
`\getmaxpar` of it is not in the aux-file.

```

1048 <*body>
1049 \newcommand*\newmaxpar}[3]{%
1050   \begingroup
1051     \expandafter\let\csname #1@Clauseformat\endcsname\@firstofone
1052     \protected@edef\@tempa{#2}\@onelevel@sanitize\@tempa
1053     \expandafter\xdef\csname max@#1@\@tempa\endcsname{#3}%
1054   \endgroup
1055 }
1056 \newcommand*\getmaxpar}[3]{%
1057   \begingroup
1058     \expandafter\let\csname #2@Clauseformat\endcsname\@firstofone
1059     \protected@edef\@tempa{#3}%
1060     \@onelevel@sanitize\@tempa
1061     \expandafter\ifx \csname max@#2@\@tempa\endcsname\relax
1062       \edef\@tempa{\endgroup\edef\noexpand#1{\expandafter\the\value{par}}}%
1063     \else
1064       \edef\@tempa{\endgroup
1065         \edef\noexpand#1{\csname max@#2@\@tempa\endcsname}}%
1066     \fi
1067   \@tempa
1068 }

```

Because some users remove `contract` from their documents without deleting the `aux`-file, we add a fallback definition of `\newmaxpar` to the `aux`-file. This avoids error messages because of undefined `\newmaxpar`.

```

1069 \AtBeginDocument{%
1070   \if@filesw
1071     \immediate\write\@auxout{%
1072       \string\providecommand*\string\newmaxpar[3]{}
1073     }%
1074   \fi
1075 }
1076 </body>

```

References

[BB24] Javier Bezos López and Johannes L. Braams. *babel* — *Multilingual support for L<sup>A</sup>T<sub>E</sub>X, LuaL<sup>A</sup>T<sub>E</sub>X, X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X and Plain T<sub>E</sub>X*. Version 24.1. This package manages culturally-determined typographical (and other) rules for a wide range of languages. A document may select a single language to be supported, or it may select several, in which case the document may switch from one language to another in a variety of ways. Babel uses contributed configuration files that provide the detail of what has to be done for each language, as well as .ini files for about 300 languages from around the World, including many written in non-Latin and RTL scripts. Many of them work with pdfL<sup>A</sup>T<sub>E</sub>X, as well as with X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X and LuaL<sup>A</sup>T<sub>E</sub>X, out of the box. A few even work with plain formats. Jan. 7, 2024. URL: <https://ctan.org/pkg/babel> (visited on 01/08/2024).

[Koh20a] Markus Kohm. *KOMA-Script. Eine Sammlung von Klassen und Paketen für L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>*. 7. Aufl. Edition DANTE. Print-Ausgabe. Berlin: Lehmanns Media, 2020. ISBN: 978-3-96543-097-6.

[Koh20b] Markus Kohm. *KOMA-Script. Eine Sammlung von Klassen und Paketen für L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub>*. 7. Aufl. Edition DANTE. eBook-Ausgabe. Berlin: Lehmanns Media, 2020. ISBN: 978-3-96543-103-4.

[Koh23a] Markus Kohm. *KOMA-Script. Die Anleitung*. 16. Juni 2023. URL: <http://mirrors.ctan.org/macros/latex/contrib/koma-script/scrguide-de.pdf> (besucht am 04.07.2023).

[Koh23b] Markus Kohm. *KOMA-Script. The Guide*. June 16, 2023. URL: <http://mirrors.ctan.org/macros/latex/contrib/koma-script/scrguide-en.pdf> (visited on 07/14/2023).

[Koh23c] Markus Kohm. *KOMA-Script — A bundle of versatile classes and packages*. Version 3.41. The KOMA-Script bundle provides replacements for the article, report, and book classes with emphasis on typography and versatility. There is also a letter class. July 7, 2023. URL: <https://ctan.org/pkg/koma-script> (visited on 07/14/2023).

[MFP21] Frank Mittelbach, Robin Fairbairns, and H. Partl. *parskip* — *Layout with zero \parindent, non-zero \parskip*. Version 2.0h. Simply changing \parskip and \parindent leaves a layout that is untidy; this package (though it is no substitute for a properly-designed class) helps alleviate this untidiness. Mar. 14, 2021. URL: <https://www.ctan.org/pkg/parskip> (visited on 01/19/2024).

[Mit21] Frank Mittelbach. *The parskip package*. Mar. 14, 2021. URL: <http://mirrors.ctan.org/macros/latex/contrib/parskip/parskip.pdf> (visited on 01/19/2024).

Change History

scrjura-v0.5b – 2010/04/05	scrjura-v0.5c – 2010/04/26
	<code>\contract@everypar:</code>
<code>juratitlepagebreak: new</code> . . . . . 22	<code>\contract@Clauseformat</code> expands

while writing to its own argument . . .	37	<code>\paragraphformat</code> : redefinition of	
first argument of <code>\newmaxpar</code> and		<code>\paragraphformat</code> . . . . .	27
second argument of <code>\getmaxpar</code> is		scrjura-v0.7 – 2013/04/18	
<code>\contract</code> not <code>\contractpars</code> . . . . .	37	<code>\contract@everypar</code> : using counter	
<code>\getmaxpar</code> : <code>\#2@Clauseformat</code>		<code>contractAbsoluteClause</code> . . . . .	37
expands to its argument . . . . .	46	<code>\contract@paragraph</code> : increase counter	
<code>\protected@edef</code> replaced by <code>\edef</code> . . . . .	46	<code>contractAbsoluteClause</code> . . . . .	31
scrjura-v0.5d – 2010/04/28		<code>\contract@subparagraph</code> : increasing	
<code>contract</code> : defining <code>\jura@env@type</code> . . . . .	27	counter <code>contractAbsoluteClause</code> . . . . .	33
missing error message added . . . . .	27	<code>contractAbsoluteClause</code> : new	
not self-redefining any more . . . . .	27	(internal) counter for all clauses to	
<code>\contract@usetype</code> : new . . . . .	37	make it possible to reset the users	
<code>\contract@env@type</code> : new . . . . .	26	clause counter . . . . .	27
<code>\contract@usetype</code> : new . . . . .	37	scrjura-v0.7 – 2013/04/28	
<code>\p@par</code> : added <code>\jura@usetype</code> with		<code>\contract@paragraph</code> : distance behind	
argument <code>\jura@env@type</code> . . . . .	37	the number is part of the number . . . . .	32
<code>\parciteformat</code> : argument moved . . . . .	44	<code>\contract@subparagraph</code> : distance after	
<code>\sentenceciteformat</code> : argument moved . . . . .	44	number . . . . .	33
scrjura-v0.5d – 2010/06/07		scrjura-v0.7 – 2013/05/02	
ref: new values <code>nopar</code> , <code>nosentence</code> ,		<code>\ellipsispar</code> : new . . . . .	26
<code>OnlyParagraph</code> . . . . .	24	<code>\parellipsis</code> : new . . . . .	26
<code>sentencecitename</code> : deprecated . . . . .	24	scrjura-v0.7 – 2013/05/23	
scrjura-v0.5e – 2011/08/31		<code>\contract@everypar</code> : initializing	
<code>\Clausemark</code> : support for		sentence number for manually	
<code>\MakeMarkcase</code> . . . . .	23	numbered paragraphs to 0 instead of	
scrjura-v0.6 – 2011/09/29		1, because first action of <code>\Sentence</code>	
<code>\cpar@indent</code> : new . . . . .	22	is increasing the number . . . . .	37
<code>\if@juratotoc</code> : replaced by counter . . . . .	21	recognize paragraph number of	
<code>juratocindent</code> : new . . . . .	22	<code>\thisparnumber</code> . . . . .	37
<code>juratoclevel</code> : definition moved . . . . .	21	<code>\contract@special@reset@par</code> : new . . . . .	38
<code>parnumber</code> : new . . . . .	22	<code>\thisparnumber</code> : new . . . . .	38
<code>\parnumberfalse</code> : moved the definition		scrjura-v0.7 – 2013/06/06	
into the definition of the option . . . . .	37	<code>\contract@everypar</code> : if paragraph	
scrjura-v0.6 – 2011/09/30		numbers are not completely	
<code>\getmaxpar</code> : argument no 3 is expanded		numerical always set the number . . . . .	37
using <code>\protected@edef</code> . . . . .	46	scrjura-v0.7 – 2013/06/07	
<code>noentry</code> : deprecated . . . . .	30	<code>\parformat</code> : new font element	
<code>notocentry</code> : new . . . . .	30	<code>parnumber</code> . . . . .	37
scrjura-v0.6a – 2012/10/08		scrjura-v0.7 – 2013/06/09	
<code>\contract@paragraph</code> : missing <code>\par</code>		<code>\toclevel@cpar</code> : new, because of	
added . . . . .	31	<code>hyperref</code> . . . . .	21
<code>\contract@subparagraph</code> : missing <code>\par</code>		scrjura-v0.7 – 2013/09/19	
added . . . . .	33	<code>\contract@lang@error</code> : Usage of	
scrjura-v0.6a – 2012/10/15		<code>\PackageError</code> instead of	
<code>parnumber</code> : value mistake message		<code>\PackageErrorNoLine</code> . . . . .	45
changed . . . . .	22	scrjura-v0.7 – 2013/11/04	
scrjura-v0.6b – 2013/04/16		<code>clausemark</code> : usage of renewed interface	
<code>\contract@paragraph</code> : using		with <code>\FamilyKeyState</code> . . . . .	23
<code>\contract@Clauseformat</code> for		<code>juratocindent</code> : usage of renewed	
manual numbers too . . . . .	32	interface with <code>\FamilyKeyState</code> . . . . .	22
<code>\newmaxpar</code> : <code>\#1@Clauseformat</code>		<code>juratotoc</code> : usage of renewed interface	
expands to its argument . . . . .	46	with <code>\FamilyKeyState</code> . . . . .	21



markboth: deprecated . . . . .	23	contractSubClause . . . . .	27
parnumber: usage of renewed interface with \FamilyKeyState . . . . .	22	\thecontractClause: must use counter Clause instead of contractClause . . . . .	27
ref: usage of renewed interface with \FamilyKeyState . . . . .	24	scrjura-v0.9c – 2015/05/13 \@defjuraenvironment: defining \the...Clause . . . . .	29
scrjura-v0.7a – 2014/01/28 \contract@sentence: \nobreak\hskip\z@ added to allow hyphenation of the first word after the sentence mark . . . . .	43	\contract@paragraph: \thecontractClause replaced by environment-dependent macro . . . . .	32
using \textsuperscript . . . . .	43	scrjura-v0.9e – 2015/11/03 \@gobble@IncludeInRelease: used temporary . . . . .	35
scrjura-v0.7b – 2014/11/03 \contract@usetype: \jura@usetype renamed . . . . .	37	\EndIncludeInRelease: used temporary . . . . .	35
\contract@afterheading: \jura@afterheading renamed . . . . .	34	\IncludeInRelease: used temporary . . . . .	35
no argument . . . . .	34	scrjura-v0.9e – 2015/11/04 \p@sentence: first start the paragraph then print the number . . . . .	44
\contract@env@type: \jura@env@type renamed . . . . .	26	scrjura-v0.9f – 2016/02/06 \contract@everypar: \nobreakspace replaced by \parformatseparation . . . . .	37
\contract@everypar: more flexible replacement of contractAbsoluteClause . . . . .	37	\parformatseparation: new . . . . .	37
\contract@paragraph: more flexible (environment)AbsoluteClause . . . . .	31	scrjura-v0.9f – 2016/02/24 \@defjuraenvironment: \protect\c_atsign_strnameuse instead of \csname...\endcsname . . . . .	29
\contract@separator: \jura@separator renamed . . . . .	37	\contract@paragraph: \protect\c_atsign_strnameuse instead of \csname...\endcsname . . . . .	32
\contract@used@everypar: \jura@everypar renamed . . . . .	35	scrjura-v0.9g – 2015/03/25 \getmaxpar: write to aux-file \if@files . . . . .	46
\contract@usetype: \jura@usetype renamed . . . . .	37	scrjura-v0.9g – 2016/03/25 \contract@everypar: writing to aux-file \if@files . . . . .	37
\ellipsispar: \thecontractAbsoluteClause made more flexible . . . . .	26	\ellipsispar: write to aux-file only \if@files . . . . .	26
scrjura-v0.7b – 2014/11/10 \getmaxpar: added fallback code to aux-file . . . . .	46	scrjura-v0.9h – 2016/04/11 clausemark: paragraphmark renamed to clausemark . . . . .	23
scrjura-v0.7b – 2014/11/11 General: General renaming of “Paragraph” into “Clause” . . . . .	20	scrjura-v0.9h – 2016/04/12 \DeclareNewJuraEnvironment: \c_atsign_strifnextchar replaced by \kernel@ifnextchar . . . . .	28
scrjura-v0.9 – 2014/11/04 \@defjuraenvironment: new . . . . .	28	\ref@ParN: \c_atsign_strifnextchar replaced by \kernel@ifnextchar . . . . .	41
scrjura-v0.9 – 2014/11/12 \DeclareNewJuraEnvironment: new . . . . .	28	\c_atsign_strifstar replaced by \kernel@ifstar . . . . .	41
scrjura-v0.9a – 2015/03/09 clausemark: internal value storage . . . . .	23	\refClause: \c_atsign_strifstar replaced by \kernel@ifstar . . . . .	39
juratocindent: internal value storage . . . . .	22	\refClauseN: \c_atsign_strifstar replaced by \kernel@ifstar . . . . .	40
juratotoc: internal value storage . . . . .	21		
parnumber: internal value storage . . . . .	22		
ref: internal value storage . . . . .	24		
scrjura-v0.9b – 2015/05/01 contractSubClause: must use counter SubClause instead of			

<code>\refL: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	39	<code>\cpar@indent</code> : replaced by usage of <code>\DeclareTOCStyleEntry</code> . . . . .	22
<code>\refN: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	39	<code>juratoclevel</code> : removed . . . . .	21
<code>\refPar: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	40	<code>\l@cpar</code> : usage of <code>tocbasic</code> . . . . .	36
<code>\refParL: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	41	<code>\toclevel@cpar</code> : handled by <code>tocbasic</code> . . . . .	21
<code>\refParS: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	41	scrjura-v3.27 – 2019/10/09	
<code>\refS: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	39	<code>\p@sentence</code> : adaption for L <sup>A</sup> T <sub>E</sub> X 2019-10-01 . . . . .	43
<code>\refSentence: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	42	scrjura-v3.28 – 2019/11/18	
<code>\refSentenceL: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	43	<code>\IncludeInRelease: \ifstr</code> renamed into <code>\Ifstr</code> . . . . .	35
<code>\refSentenceN: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	43	<code>parnumber: \ifstr</code> renamed into <code>\Ifstr</code> . . . . .	22
<code>\refSentenceS: \c_atsign_strifstar</code> replaced by <code>\kernel@ifstar</code> . . . . .	43	scrjura-v3.36 – 2022/01/25	
scrjura-v0.9i – 2017/02/23		<code>\@documentcontracttrue</code> : switched to a <b>KOMA-Script</b> option . . . . .	21
<code>\contract@paragraph</code> : usage of <code>\ext@toc</code> . . . . .	32	<code>contract</code> : defining an environment (because of hooks) . . . . .	27
<code>\contract@subparagraph</code> : usage of <code>\ext@toc</code> . . . . .	33	scrjura-v3.39 – 2022/11/11	
scrjura-v3.24 – 2017/05/29		<code>clausemark</code> : initial dot in member argument of option storage commands removed . . . . .	23
<code>\contract@lang@error</code> : adaption for language name bug in <b>babel</b> 3.10 . . . . .	45	<code>juratocindent</code> : initial dot in member argument of option storage commands removed . . . . .	22
scrjura-v3.25 – 2017/12/08		<code>juratotoc</code> : initial dot in member argument of option storage commands removed . . . . .	21
<code>\@defjuraenvironment</code> : <code>\reserved@defjuraenvironment</code> set back to <code>\relax</code> . . . . .	28	<code>parnumber</code> : initial dot in member argument of option storage commands removed . . . . .	22
Font alias fixed . . . . .	29	<code>ref</code> : initial dot in member argument of option storage commands removed . . . . .	24
new command options <code>ClauseFont</code> . . . . .	28	scrjura-v3.39 – 2022/11/16	
using local instead of global command options . . . . .	28	<code>markboth</code> : only with <b>KOMA-Script</b> 3 . . . . .	23
<code>\contract@paragraph@font</code> : using element <code>Clause</code> instead of <code>contract.Clause</code> . . . . .	31	<code>sentencecitename</code> : only with <b>KOMA-Script</b> 3 . . . . .	24
scrjura-v3.25 – 2017/12/19		scrjura-v3.41 – 2023/06/24	
<code>\withoutparnumber</code> : new . . . . .	37	<code>\contract@paragraph</code> : update of <code>\c_atsign_strcurrentcounter</code> added . . . . .	32
scrjura-v3.26 – 2018/07/20		<code>\contract@subparagraph</code> : update of <code>\c_atsign_strcurrentcounter</code> added . . . . .	33
<code>\p@sentence</code> : new font element <code>sentencecounter</code> . . . . .	44	<code>\theH@AbsClause</code> : Ulrike Fischer requested to use <code>\def</code> instead of <code>\newcommand*</code> . . . . .	31
new formatting <code>\sentencecounterformat</code> . . . . .	44	<code>\theHpar</code> : Ulrike Fischer requested to use <code>\def</code> instead of <code>\newcommand*</code> . . . . .	37
<code>\sentencecounterformat</code> : new . . . . .	44	<code>\theHsentence</code> : Ulrike Fischer requested to use <code>\def</code> instead of <code>\newcommand*</code> . . . . .	43
scrjura-v3.27 – 2019/02/25			
<code>\contract@afterheading</code> : <code>\CheckCommand</code> for <code>\c_atsign_strafterheading</code> . . . . .	34		

<code>\theHSubClause</code> : Ulrike Fischer requested to use <code>\def</code> instead of <code>\newcommand*</code> . . . . .	31	<code>\contract@used@everypar</code> : <code>\scrjura@everypar</code> renamed . . . . .	35
		<code>\contract@usetype</code> : <code>\scrjura@usetype</code> renamed . . . . .	37
v0.0.1 – 2023-10-10		<code>\if@contract@skiphyperref</code> : <code>\if@scrjura@skiphyperref</code> renamed . . . . .	27
General: new KOMA-Script spin-off . . .	1	v0.0.2 – 2024/01/23	
<code>\contract@usetype</code> : <code>\scrjura@usetype</code> renamed . . . . .	37	<code>\contract@paragraph</code> : the argument is optional . . . . .	31
<code>\contract@afterheading</code> : <code>\scrjura@afterheading</code> renamed . . .	34	<code>\contract@subparagraph</code> : the argument is optional . . . . .	33
<code>\contract@doendpe</code> : <code>\scrjura@doendpe</code> renamed . . . . .	35	v0.9 – 2024-02-02	
<code>\contract@env@type</code> : <code>\scrjura@env@type</code> renamed . . . . .	26	General: first release as standalone package . . . . .	1
<code>\contract@lang@error</code> : <code>\scrjura@lang@error</code> renamed . . .	45	v0.92 – 2024/06/07	
<code>\contract@separator</code> : <code>\scrjura@separator</code> renamed . . . .	37	General: need at least <code>scrkbase</code> v3.39 . .	20
<code>\contract@special@reset@par</code> : <code>\scrjura@special@par</code> renamed . . .	38	v0.92 – 2025/11/12	
<code>\scrjura@special@reset@par</code> renamed . . . . .	38	General: deprecated <code>scrjura</code> commands removed . . . . .	26
		v0.92 – 2026-04-23	
		<code>\@defjuraenvironment</code> : missing group added to <code>\the...Clause</code> . . . . .	29

## Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in *roman* refer to the code lines where the entry is used.

<b>Symbols</b>		<code>\Clause</code> . . . . .	5
<code>@AbsClause</code> ( <i>cnt.</i> ) . . . . .	449	<code>\Clauseformat</code> . . . . .	6, 269
<b>A</b>		<code>\Clausemark</code> . . . . .	91
<code>\AutoPar</code> . . . . .	578	<code>\contractSetup</code> . . . . .	3
<b>C</b>		<code>\DeclareNewJuraEnvironment</code> . . .	12, 303
<code>\Clause</code> . . . . .	5	<code>\ellipsispar</code> . . . . .	8, 248
<code>clause</code> . . . . .	2, 4	<code>\getmaxpar</code> . . . . .	1048
<code>clause</code> . . . . .	5	<code>\ManualPar</code> . . . . .	578
<code>\Clauseformat</code> . . . . .	6, 269	<code>\newmaxpar</code> . . . . .	1048
<code>\Clausemark</code> . . . . .	91	<code>\paragraphformat</code> . . . . .	269
<code>clausemark</code> ( <i>opt.</i> ) . . . . .	91	<code>\parciteformat</code> . . . . .	976
<code>clausemark=&lt;value&gt;</code> ( <i>opt.</i> ) . . . . .	7	<code>\parellipsis</code> . . . . .	8, 248
<code>clausemark=both</code> ( <i>opt.</i> ) . . . . .	7	<code>\parformat</code> . . . . .	8, 695
<code>clausemark=false</code> ( <i>opt.</i> ) . . . . .	7	<code>\parformatseparation</code> . . . . .	8, 695
<code>clausemark=forceboth</code> ( <i>opt.</i> ) . . . . .	7	<code>\parlongformat</code> . . . . .	1003
<code>clausemark=forceright</code> ( <i>opt.</i> ) . . . . .	7	<code>\parname</code> . . . . .	12, 1010
<code>clausemark=right</code> ( <i>opt.</i> ) . . . . .	7	<code>\parnumericformat</code> . . . . .	1003
Commands:		<code>\parshortformat</code> . . . . .	1003
<code>\AutoPar</code> . . . . .	578	<code>\parshortname</code> . . . . .	1010
		<code>\partshortname</code> . . . . .	12

<code>\ref</code> .....	9	<code>dummy (opt.)</code> .....	444
<code>\refClause</code> .....	9, 793		
<code>\refClauseN</code> .....	9, 822	<b>E</b>	
<code>\refL</code> .....	9, 768	<code>\ellipsispar</code> .....	8, 248
<code>\refN</code> .....	9, 785	<code>entry (opt.)</code> .....	417
<code>\refPar</code> .....	10, 831	Environments:	
<code>\refParL</code> .....	10, 866	<code>contract</code> .....	5, 269
<code>\refParN</code> .....	10, 879		
<code>\refParS</code> .....	10, 876	<b>G</b>	
<code>\refS</code> .....	9, 777	<code>\getmaxpar</code> .....	1048
<code>\refSentence</code> .....	10, 896		
<code>\refSentenceL</code> .....	10, 931	<b>H</b>	
<code>\refSentenceN</code> .....	10, 944	<code>head (opt.)</code> .....	417
<code>\refSentenceS</code> .....	10, 941		
<code>\Sentence</code> .....	9	<b>J</b>	
<code>\sentenceciteformat</code> .....	990	<code>juratitlepagebreak (opt.)</code> .....	7, 63
<code>\sentenceclongformat</code> .....	1003	<code>juratitlepagebreak=&lt;boolean&gt; (opt.)</code> ....	7
<code>\sentencename</code> .....	12, 1010	<code>juratocindent (opt.)</code> .....	49
<code>\sentencenumberformat</code> .....	9, 973	<code>juratocindent=&lt;length&gt; (opt.)</code> .....	4
<code>\sentencenumericformat</code> .....	1003	<code>juratoclevel (cnt.)</code> .....	33
<code>\sentenceshortformat</code> .....	1003	<code>juratocnumberwidth (opt.)</code> .....	49
<code>\sentenceshortname</code> .....	12, 1010	<code>juratocnumberwidth=&lt;length&gt; (opt.)</code> .	4, 15
<code>\SubClause</code> .....	5	<code>juratotoc (opt.)</code> .....	4, 33
<code>\thecontractClause</code> .....	269	<code>juratotoc=&lt;boolean&gt; (opt.)</code> .....	4
<code>\thecontractSubClause</code> .....	269	<code>juratotoc=&lt;integer&gt; (opt.)</code> .....	4
<code>\theHClause</code> .....	449		
<code>\theHpar</code> .....	695	<b>K</b>	
<code>\theHsentence</code> .....	948	KOMA-Script font elements:	
<code>\theHSubClause</code> .....	449	<code>contract.Clause</code> .....	446
<code>\thepar</code> .....	8, 695	<code>sentence.number</code> .....	973
<code>\thesentence</code> .....	9, 948		
<code>\thisparnumber</code> .....	759	<b>L</b>	
<code>\withoutparnumber</code> .....	8, 695	Lengths (internal):	
<code>contract</code> .....	2, 5	<code>\cpar@indent</code> .....	49
<code>contract (env.)</code> .....	5, 269	<code>\cpar@numberwidth</code> .....	49
<code>contract (opt.)</code> .....	5, 19		
<code>contract.Clause (font)</code> .....	446	<b>M</b>	
<code>contractAbsoluteClause (cnt.)</code> .....	269	<code>\ManualPar</code> .....	578
<code>contractClause (cnt.)</code> .....	269	<code>markboth (opt.)</code> .....	91
<code>\contractSetup</code> .....	3	<code>markright (opt.)</code> .....	91
<code>contractSubClause (cnt.)</code> .....	269		
Counters:		<b>N</b>	
<code>@AbsClause</code> .....	449	<code>\newmaxpar</code> .....	1048
<code>contractAbsoluteClause</code> .....	269	<code>noentry (opt.)</code> .....	417
<code>contractClause</code> .....	269	<code>nohead (opt.)</code> .....	417
<code>contractSubClause</code> .....	269	<code>notocentry (opt.)</code> .....	417
<code>juratoclevel</code> .....	33	<code>number (opt.)</code> .....	439
<code>par</code> .....	8, 695		
<code>sentence</code> .....	9, 948	<b>O</b>	
		Options:	
<b>D</b>		<code>clausemark</code> .....	91
<code>\DeclareNewJuraEnvironment</code> .....	12, 303	<code>clausemark=&lt;value&gt;</code> .....	7
		<code>clausemark=both</code> .....	7
		<code>clausemark=false</code> .....	7
		<code>clausemark=forceboth</code> .....	7

clausemark=forceright	7	omission	8
clausemark=right	7	\paragraphformat	269
contract	5, 19	paragraphmark ( <i>opt.</i> )	91
dummy	444	\parciteformat	976
entry	417	parcitename ( <i>opt.</i> )	153
head	417	\parellipsis	8, 248
juratitlepagebreak	7, 63	\parformat	8, 695
juratitlepagebreak=< <i>boolean</i> >	7	\parformatseparation	8, 695
juratocindent	49	\parlongformat	1003
juratocindent=< <i>length</i> >	4	\parname	12, 1010
juratocnumberwidth	49	parnumber ( <i>opt.</i> )	7, 66
juratocnumberwidth=< <i>length</i> >	4, 15	parnumber= <i>value</i> ( <i>opt.</i> )	7
juratotoc	4, 33	parnumber=auto ( <i>opt.</i> )	7, 8
juratotoc=< <i>boolean</i> >	4	parnumber=false ( <i>opt.</i> )	7
juratotoc=< <i>integer</i> >	4	parnumber>manual ( <i>opt.</i> )	8
markboth	91	parnumber=true ( <i>opt.</i> )	7
markright	91	\parnumericformat	1003
noentry	417	\parshortformat	1003
nohead	417	\parshortname	1010
notocentry	417	parskip ( <i>opt.</i> )	7
number	439	\partshortname	12
paragraphmark	91	postskip ( <i>opt.</i> )	440
parcitename	153	preskip ( <i>opt.</i> )	440
parnumber	7, 66		
parnumber=< <i>value</i> >	7		
parnumber=auto	7, 8	R	
parnumber=false	7	\ref	9
parnumber>manual	8	ref ( <i>opt.</i> )	153
parnumber=true	7	ref=< <i>value</i> > ( <i>opt.</i> )	10
parskip	7	ref=long ( <i>opt.</i> )	11, 11
postskip	440	ref=numeric ( <i>opt.</i> )	11
preskip	440	ref=parlong ( <i>opt.</i> )	11, 11
ref	153	ref=parnumeric ( <i>opt.</i> )	11
ref=< <i>value</i> >	10	ref=paroff ( <i>opt.</i> )	11
ref=long	11, 11	ref=parshort ( <i>opt.</i> )	11
ref=numeric	11	ref=sentencenumeric ( <i>opt.</i> )	11
ref=parlong	11, 11	ref=sentenceoff ( <i>opt.</i> )	11
ref=parnumeric	11	ref=senceshort ( <i>opt.</i> )	11
ref=paroff	11	ref=value ( <i>opt.</i> )	11
ref=parshort	11	\refClause	9, 793
ref=sentencenumeric	11	\refClauseN	9, 822
ref=sentenceoff	11	\refL	9, 768
ref=senceshort	11	\refN	9, 785
ref=value	11	\refPar	10, 831
sentencecitename	153	\refParL	10, 866
title	417	\refParN	10, 879
tocentry	417	\refParS	10, 876
		\refS	9, 777
P		\refSentence	10, 896
par ( <i>cnt.</i> )	8, 695	\refSentenceL	10, 931
paragraph	see clause	\refSentenceN	10, 944
numbering	7	\refSentenceS	10, 941

<b>S</b>	
section	<i>see</i> <a href="#">clause</a>
\Sentence	<a href="#">9</a>
sentence	
number	<a href="#">9</a>
sentence (cnt.)	<a href="#">9</a> , <a href="#">948</a>
\sentenceciteformat	<a href="#">990</a>
sentencecitename (opt.)	<a href="#">153</a>
\sentencelongformat	<a href="#">1003</a>
\sentencename	<a href="#">12</a> , <a href="#">1010</a>
sentence number (font)	<a href="#">973</a>
\sentence numberformat	<a href="#">9</a> , <a href="#">973</a>
\sentence numericformat	<a href="#">1003</a>
\sentence shortformat	<a href="#">1003</a>
\sentence shortname	<a href="#">12</a> , <a href="#">1010</a>
\SubClause	<a href="#">5</a>
<b>T</b>	
T <sub>E</sub> X macros (internal):	
\@defjuraenvironment	<a href="#">322</a>
\@documentcontractfalse	<a href="#">19</a>
\@documentcontracttrue	<a href="#">19</a>
\@doendpe	<a href="#">622</a>
\@gobble@IncludeInRelease	<a href="#">622</a>
\@juratitlepagebreakfalse	<a href="#">63</a>
\@juratitlepagebreaktrue	<a href="#">63</a>
\contract@usetype	<a href="#">691</a>
\contract@afterheading	<a href="#">587</a>
\contract@Clauseformat	<a href="#">269</a>
\contract@doendpe	<a href="#">622</a>
\contract@env@type	<a href="#">245</a>
\contract@everypar	<a href="#">695</a>
\contract@lang@error	<a href="#">1010</a>
\contract@paragraph	<a href="#">417</a>
\contract@paragraph@font	<a href="#">446</a>
\contract@postskip	<a href="#">440</a>
\contract@preskip	<a href="#">440</a>
\contract@sentence	<a href="#">948</a>
\contract@separator	<a href="#">688</a>
\contract@special@par	<a href="#">759</a>
\contract@special@reset@par	<a href="#">759</a>
\contract@subparagraph	<a href="#">517</a>
\contract@used@everypar	<a href="#">621</a>
\contract@usetype	<a href="#">691</a>
\EndIncludeInRelease	<a href="#">622</a>
\if@contract@skiphyperref	<a href="#">269</a>
\if@documentcontract	<a href="#">19</a>
\if@juratitlepagebreak	<a href="#">63</a>
\if@juratotoc	<a href="#">33</a>
\ifcontract@dummy	<a href="#">444</a>
\ifparnumber	<a href="#">695</a>
\IncludeInRelease	<a href="#">622</a>
\l@cpar	<a href="#">681</a>
\p@par	<a href="#">695</a>
\p@sentence	<a href="#">948</a>
\par@cite	<a href="#">976</a>
\parcite@format	<a href="#">153</a>
\parnumberfalse	<a href="#">695</a>
\parnumbertrue	<a href="#">695</a>
\ref@ParN	<a href="#">879</a>
\ref@Clause	<a href="#">793</a>
\ref@ClauseN	<a href="#">822</a>
\ref@L	<a href="#">768</a>
\ref@N	<a href="#">785</a>
\ref@Par	<a href="#">831</a>
\ref@ParN	<a href="#">879</a>
\ref@ParX	<a href="#">866</a>
\ref@S	<a href="#">777</a>
\ref@Sentence	<a href="#">896</a>
\ref@SentenceX	<a href="#">931</a>
\sentence@cite	<a href="#">990</a>
\sentencecite@format	<a href="#">153</a>
\theH@AbsClause	<a href="#">449</a>
\toclevel@cpar	<a href="#">33</a>
\thecontractClause	<a href="#">269</a>
\thecontractSubClause	<a href="#">269</a>
\theHClause	<a href="#">449</a>
\theHpar	<a href="#">695</a>
\theHsentence	<a href="#">948</a>
\theHSubClause	<a href="#">449</a>
\thepar	<a href="#">8</a> , <a href="#">695</a>
\thesentence	<a href="#">9</a> , <a href="#">948</a>
\thisparnumber	<a href="#">759</a>
title (opt.)	<a href="#">417</a>
tocentry (opt.)	<a href="#">417</a>
<b>W</b>	
\withoutparnumber	<a href="#">8</a> , <a href="#">695</a>