

The `cleveref-usedon` package ^{*}

Sven Pistre [†]

Released 2023-04-21

Abstract

This package adds “forward-referencing” to the `cleveref` package. Any label can be referenced with the new optional argument `UsedOn` passed to `\cref`. Doing so, will print an info message at the original label location (in a theorem environment, say) which reads “*Used on pages* *<list of pages>*.”. This functionality is complementary to `hyperref`’s `pagebackref` or `bibtex`’s `backref` option for the bibliography. It might be useful for authors of longer texts such as textbooks or theses, where a lot of supplementary results and information are given in early chapters, appendices or exercises. The message on which pages these results will be used can be a helpful information for the reader of the final text. Additionally, a bug in `cleveref` v0.21.4 is patched.

Contents

| | | |
|----------|---|----------|
| 1 | Introduction | 2 |
| 2 | Usage | 3 |
| 2.1 | The option <code>[\UsedOn]</code> | 4 |
| 2.2 | The experimental options <code>[\UsedBy]</code> and <code>[\UsedByAndOn]</code> | 4 |
| 3 | Hints and tips | 5 |
| 3.1 | Editing the info messages | 6 |
| 4 | Interaction with other packages | 7 |
| 5 | Future features | 7 |
| 6 | Implementation | 8 |
| 6.1 | Options and requirements | 8 |
| 6.2 | Patches of known bugs to <code>cleveref</code> | 9 |
| 6.3 | Overloading of label and <code>cref</code> | 9 |

^{*}This document corresponds to `cleveref-usedon` v0.4.0, last revised 2023-04-21.

[†]E-mail: cleveref-usedon@sven-pistre.com

1 Introduction

Imagine you are reading a long mathematical text such as a text book or a thesis. There are plenty of supplementary lemmas, propositions, theorems and/or exercises throughout the whole text. You ask yourself “Gosh, while Lemma 1.12 is certainly an interesting result *where* is this result used later on in this long text? I really would find that helpful to decide *why* I should read the proof.” You can, of course, use the PDF search function of your viewer to look up the string “Lemma 1.12” but wouldn’t it be more helpful if Lemma 1.12 already indicates all or at least its most useful/crucial applications via an info message?

This is what the package `cleveref-usedon` tries to address. The info message “*Used on p. 40, 43-45 and 101.*” would then be printed to the header of Lemma 1.12. For example, we have given the following theorem the label

```
\label{thm:SqrtTwoIrrational}.
```

Theorem 1.1. (Used on pages 2 and 7.)

The number $\sqrt{2}$ is irrational.

Now we can reference this theorem via

```
\cref[UsedOn]{thm:SqrtTwoIrrational}:
```

A proof of Theorem 1.1 can be traced back to Euclid.

We will now reference this theorem without the optional argument `[UsedOn]`. So let’s clear the page of this PDF, so that we can see the effects of calling

```
\cref{thm:SqrtTwoIrrational}
```

more clearly.

Note that the current page number 3 is not included in the list of page references in the header of Theorem 1.1.

2 Usage

The `cleveref-usedon` package uses `cleveref` v0.21.4 as its base. To freely cite from the `cleveref` documentation:

The `cleveref-usedon` package is loaded in the usual way, by putting the line

```
\usepackage{cleveref-usedon}
```

in your document's preamble. However, care must be taken when using `cleveref` in conjunction with other packages that modify L^AT_EX's referencing system (see Section 13 of `cleveref`'s documentation). Basically, `cleveref-usedon` must be loaded *last* but definitely AFTER `hyperref`.

Available package options includes `UsedOn`, `UsedBy`, `UsedByAndOn` and the lower-cased or abbreviated version of them. They are used for setting the default behavior in the document. For example, if you use the package option `UsedOn`, then unless explicitly specified otherwise, every `\cref`, `\Cref` and `\labelcref` would automatically use the mode `UsedOn` (see below).

| | |
|-------------------------|--|
| <code>\cref</code> | <code>\cref[<i><Option></i>]{<i><LabelName></i>}</code> |
| <code>\Cref</code> | <code>\Cref[<i><Option></i>]{<i><LabelName></i>}</code> |
| <code>\labelcref</code> | <code>\labelcref[<i><Option></i>]{<i><LabelName></i>}</code> |

The `\cref` macro can be called with options `UsedOn` (see Section 2.1), `UsedBy` (experimental, see Section 2.2) and `UsedByAndOn` (experimental, see Section 2.2) or their short forms `uo`, `ub`, `ubao`. This is case-insensitive, i.e. you could also write¹

```
\cref[UsEdOn]{<LabelName>},
\cref[uO]{<LabelName>}.
```

The package `cleveref-usedon` is implemented using the L^AT_EX3 programming layer `expl3`. If you are interested, I have spent some time to document and comment on the implementation in Section 6. On an abstract level the implementation is as follows: Whenever the label `<LabelName>` gets referenced with one of the options at some location via `\cref[<Option>]{<LabelName>}`, an additional auxiliary label is created at this very location. This auxiliary label has the form `<Option>@<LabelName>@<Counter>` where `<Counter>` is an integer that counts how often the label `<LabelName>` has been referenced with `<Option>`. At the end of the L^AT_EX run, the final value of this counter is written to the `.aux` file as a key-value pair:

$$\langle Option \rangle @ \langle LabelName \rangle = \langle MaxCounter \rangle$$

In the second L^AT_EX run, we read this counter from the `.aux` file. Then, at the original location of the referenced label `<LabelName>`, we can now pass the list of auxiliary labels

$$\langle Option \rangle @ \langle LabelName \rangle @ 1, \dots, \langle Option \rangle @ \langle LabelName \rangle @ \langle MaxCounter \rangle$$

to `\cpageref` (and `\cref` for the experimental options) and write the forward-referencing info message.

¹But why would you want to?

2.1 The option [`\UsedOn`]

`\UsedOn` This option adds the message

(Used on page(s) (list of page(s)).)

The text is followed by a line break and is set after the original location of the referenced label `\LabelName`. If `hyperref` has been loaded, there will also be hyperlinks to the corresponding pages from where the label has been referenced.

If the original label has been set in a theorem-like environment such as

```
\begin{theorem} \label{thm:SqrtTwoIrrational}
  The number  $\sqrt{2}$  is irrational.
\end{theorem}
```

then the info message is printed in the header of this theorem-like environment. The same functionality can be used for `\Cref`.

2.2 The experimental options [`\UsedBy`] and [`\UsedByAndOn`]

`\UsedBy`
`\UsedByAndOn` The option [`\UsedBy`] adds the message

(Used by (list of theorem-like destination(s)).)

The option [`\UsedByAndOn`] adds the message

(Used by (list of theorem-like destination(s)) on page(s) (list of page(s)).)

Each text is followed by a line break and is set after the original location of the referenced label `\LabelName`. If `hyperref` has been loaded, there will also be hyperlinks to the destinations.

For example, suppose we have the following lemma.

Lemma 2.1. (Used by Corollary 2.2.)

Any smooth function $f: \mathbb{R} \rightarrow \mathbb{R}$ is continuous.

And we will use it in the proof of the following result.

Corollary 2.2. (Used by Corollary 2.3 on page 4.)

Suppose $f: \mathbb{R} \rightarrow \mathbb{R}$ is smooth. The derivative $f': \mathbb{R} \rightarrow \mathbb{R}$ is continuous.

Proof. The derivative of a smooth map is itself smooth. Hence, the claim follows by Lemma 2.1. □

The previous result will in turn be used in the proof of the next one.

Corollary 2.3. *Suppose $f: \mathbb{R} \rightarrow \mathbb{R}$ is smooth and $k \in \mathbb{N}$. The k th derivative $f^{(k)}: \mathbb{R} \rightarrow \mathbb{R}$ is continuous.*

Proof. This follows from Corollary 2.2 by induction. □

The code for the above examples is as follows:

```

\begin{lemma}          \label{lemma:SmoothFunction}
  Any smooth function  $f\colon \mathbb{R}\to \mathbb{R}$ 
  is continuous.
\end{lemma}

\begin{corollary}      \label{cor:DerivativeContinuous}
  Suppose  $f\colon \mathbb{R}\to \mathbb{R}$  is smooth.
  The derivative  $f'\colon \mathbb{R}\to \mathbb{R}$ 
  is continuous.
\begin{proof}
  The derivative of a smooth map is itself smooth.
  Hence, the claim follows by \cref[UsedBy]{lemma:SmoothFunction}.
\end{proof}
\end{corollary}

\begin{corollary}      \label{cor:AllDerivativesContinuous}
  Suppose  $f\colon \mathbb{R}\to \mathbb{R}$  is smooth and
   $k\in\mathbb{N}$ . The  $k$ th derivative
   $f^{(k)}\colon \mathbb{R}\to \mathbb{R}$ 
  is continuous.
\begin{proof}
  This follows from
  \cref[UsedByAndOn]{cor:DerivativeContinuous} by induction.
\end{proof}
\end{corollary}

```

Unfortunately, due to how this package is currently implemented, to get these experimental options to work it is necessary to abuse the usage of proof environments. Namely, one needs to nest the proof environment *inside* the theorem-like environment. Note carefully how the proof environments are (ab)used in the above code example.

This is – as far as I know – not how these environments are supposed to be used. In particular, placing text between theorem-like environment and the corresponding proof, as is often common, will result in a wrong reference. Namely, instead of referencing the theorem-like environment by name only the corresponding section name would be printed, e.g. “*Used by Section 2.2*”. You can see this for yourself, if you move the proof environment out of the theorem-like environment in the above examples. Hence, using proof environments correctly results in messages which are less helpful to the reader. On the other hand, using this experimental functionality to help the reader forces users (i.e. authors) of this package to use proof environments incorrectly. This sounds like a No-Free-Lunch theorem... Therefore, use these two experimental options at your own discretion!

3 Hints and tips

If you use the `capitalise` option for `cleveref`, you might want to revert this capitalisation for page references for more visual appeal by putting

```
\crefname{page}{page}{pages}
```

in your document’s preamble, after loading `cleveref-usedon`.

It is recommended to not use the optional arguments for equation-like environments such as Eq. (1) because sometimes² the info message will — unhelpfully — be printed inside the equation environment, like so (this might or might not³ show undesired behaviour):

$$\int_M d\omega = \int_{\partial M} \omega. \quad (1)$$

So, one should use this functionality only for theorem-like environments such as theorems, lemmas and exercises etc.

If one references the same label multiple times but with different options, say `UsedOn` and `UsedBy`, then *both* info messages are printed after the original label location. This is not how this functionality was intended and you shouldn’t use it like that. I am not going to implement a check which various combinations of these options are used for the same label.

3.1 Editing the info messages

| | |
|----------------------------------|---|
| <code>\UsedOnMessage</code> | <code>\UsedOnMessage{⟨PageList from cpageref⟩}</code> |
| <code>\UsedByMessage</code> | <code>\UsedByMessage{⟨EnvironmentList from cpageref⟩}</code> |
| <code>\UsedByAndOnMessage</code> | <code>\UsedByAndOnMessage{⟨EnvironmentList from cpageref⟩}{⟨PageList from cpageref⟩}</code> |

The standard messages which get printed to the first line of the labelled environment are

(Used on ⟨PageList⟩.),
(Used by ⟨EnvironmentList⟩.),
(Used by ⟨EnvironmentList⟩) on ⟨PageList⟩.),

respectively — followed by a line break — where `⟨PageList⟩` is generated internally by `cleveref` via `\cpageref` and `⟨EnvironmentList⟩` is generated internally by `cleveref` via `\cref`. You can change these behaviours by redefining the macros `\UsedOnMessage`, `\UsedByMessage` and `\UsedByAndOnMessage`, e.g. as

```
\RenewDocumentCommand \UsedOnMessage { m }
{
  \emph{\UsedOnText{#1}} \\[.3\baselineskip]
}
\RenewDocumentCommand \UsedByMessage { m }
{
  \emph{\UsedByText{#1}} \\[.3\baselineskip]
}
\RenewDocumentCommand \UsedByAndOnMessage { m }
{
  \emph{\UsedByAndOnText{#1}} \\[.3\baselineskip]
}
```

²I haven’t quite tracked down this bug.

³In version 0.2.0 of this package, the text “*Used on page 2.*” was printed right after the formula in the equation environment.

| | |
|----------------------------------|---|
| <code>\SetUsedOnText</code> | <code>\SetUsedOnText{<Text with #1>}</code> |
| <code>\SetUsedByText</code> | <code>\SetUsedByText{<Text with #1>}</code> |
| <code>\SetUsedByAndOnText</code> | <code>\SetUsedByAndOnText{<Text with #1 and #2>}</code> |

You may redefine the text of the standard messages via these three macros. As an example, the default text for `UsedOn` is defined via

```
\SetUsedOnText
{
  \IfLanguageName{english}
  { (Used~on~#1.) } {}
  \IfLanguageName{french}
  { (Apparaît~en~#1.) } {}
  \IfLanguageName{ngerman}
  { (Wird~auf~#1.) } {}
  \IfLanguageName{spanish}
  { (Aparece~en~#1.) } {}
}
```

4 Interaction with other packages

All interactions with other packages mentioned in Section 13 of `cleveref`'s documentation also apply to `cleveref-usedon`. In fact (if `cleveref-usedon` is loaded last), `ntheorem`'s `\thref` and `varioref`'s `\vref` also obtain the additional `UsedOn` functionality because `cleveref` redefines these macros to be aliases for `\cref`. Of course, they need to be loaded in the correct order, i.e.

```
\usepackage{ntheorem}
\usepackage{hyperref}
\usepackage{cleveref-usedon}
```

or

```
\usepackage{varioref}
\usepackage{hyperref}
\usepackage{cleveref-usedon}
```

5 Future features

For all feature requests, either [create a github issue](#) or [send me an email](#).

Let's just reference Theorem [1.1](#) one last time for the fun of it, check page [2](#) again to see the effect to the reference list in the header of Theorem [1.1](#).

6 Implementation

Start the DocStrip guards.

```

1 <*package>
   Identify the internal prefix (LATEX3 DocStrip convention).
2 <@@=UsedOn>
   If the LATEX version is too old, then abort loading and show an error message. The
   macro \IfFormatAtLeastTF is not be available on LATEX versions older than 2020-10-01
   so we need to provide an internal workaround.
3 \providecommand\IfFormatAtLeastTF{\@ifl@t@r\fmtversion}
4 \IfFormatAtLeastTF{2021-06-01}{%
5   % LaTeX2e version new enough
6 }{%
7   \PackageError{cleveref-usedon}{%
8     Mismatched~LaTeX~support~files~detected.\MessageBreak
9     Your~LaTeX~format~is~dated~\fmtversion,\MessageBreak
10    but~the~package~cleveref-usedon~\MessageBreak
11    requires~at~least~2021-06-01.\MessageBreak
12    Update~your~TeX~distribution.\MessageBreak
13    \MessageBreak
14    Loading~cleveref-usedon~will~abort!}%
15    {Update~your~TeX~distribution~using~your~TeX~package~manager.}%
16 }

```

If the version of the L^AT_EX3 kernel is too old, then abort loading and show an error message. The macro \IfExplAtLeastTF currently does not exist at all⁴, but we can provide an internal workaround.

```

17 \providecommand\IfExplAtLeastTF{\@ifl@t@r\ExplLoaderFileDate}
18 \RequirePackage{expl3}[2021-05-16]
19 \IfExplAtLeastTF{2021-05-16}{%
20   % expl3 version new enough
21 }{%
22   \PackageError{cleveref-usedon}{%
23     Support~package~expl3~too~old.\MessageBreak
24     The~L3~programming~layer~in~the~LaTeX~format~\MessageBreak
25     is~dated~\ExplLoaderFileDate,\MessageBreak
26     but~the~package~cleveref-usedon~\MessageBreak
27     requires~at~least~2021-05-16.\MessageBreak
28     Update~your~TeX~distribution.\MessageBreak
29     \MessageBreak
30     Loading~cleveref-usedon~will~abort!}%
31     {Update~your~TeX~distribution~using~your~TeX~package~manager.}%
32 }

```

6.1 Options and requirements

Below are the available package options.

```

33
34 \keys_define:nn { cleveref-usedon }
35 {

```

⁴But might in the future, see [github issue #1004](#).


```

36 , Default      .tl_set:N    = \l__UsedOn_default_tl
37 , Default      .initial:n   = { }
38 , Default      .value_required:n = true
39 , UsedOn        .meta:n      = { Default = UsedOn }
40 , usedon        .meta:n      = { Default = UsedOn }
41 , uo           .meta:n      = { Default = UsedOn }
42 , UsedBy        .meta:n      = { Default = UsedBy }
43 , usedby        .meta:n      = { Default = UsedBy }
44 , ub           .meta:n      = { Default = UsedBy }
45 , UsedByAndOn   .meta:n      = { Default = UsedByAndOn }
46 , usedbyandon   .meta:n      = { Default = UsedByAndOn }
47 , ubao          .meta:n      = { Default = UsedByAndOn }
48

```

All other package options get passed on to `cleveref` which is the base for the current package here.

```

49 , unknown      .code:n      =
50   { \PassOptionsToPackage { \CurrentOption } { cleveref } }
51 }
52 \ProcessKeyOptions [ cleveref-usedon ]
53
54 \RequirePackage{cleveref}[2018/03/27]

```

6.2 Patches of known bugs to `cleveref`

The following fixes the range bug for `\cpageref` in `cleveref` v0.21.4. See <https://tex.stackexchange.com/a/620066/267438>. (We need to temporarily reset the DocStrip guards.)

```

55 <@@=>
56 \newcommand*{\@setcpagerefrange}[3]{%
57   \@setcpagerefrange{#1}{#2}{cref}{#3}}
58 \newcommand*{\@setCpagerefrange}[3]{%
59   \@setcpagerefrange{#1}{#2}{Cref}{#3}}
60 \newcommand*{\@setlabelcpagerefrange}[3]{%
61   \@setcpagerefrange{#1}{#2}{labelcref}{#3}}
62 <@@=UsedOn>

```

6.3 Overloading of label and cref

We need variants of `\str_case:nn` which expand the input string token. This will be used to match options for `__UsedOn_Processor`.

```

63 \prg_generate_conditional_variant:Nnn \str_case:nn { x } { T, TF }
64 \cs_generate_variant:Nn \str_case:nn { x }

```

`\g__UsedOn_k_seq` Let's initialise a global key sequence for those label names that have been referenced via `[UsedOn]`, `[UsedBy]` or `[UsedByAndOn]`.

```

65 \seq_new:N \g__UsedOn_k_seq

```

(End of definition for `\g__UsedOn_k_seq`.)

`\g__UsedOn_kv_prop` And we'll also create a global key-value property list with label names as keys and the maximal amount of times they have been referenced via `[UsedOn]` as values (possibly known from the last pdf_latex run).

```

66 \prop_new:N \g__UsedOn_kv_prop

```

(End of definition for `\g__UsedOn_kv_prop`.)

`\g__UsedOn_Options_clist` This `clist` contains all options that can be passed to `\cref` which are currently implemented.

```

67 \clist_new:N \g__UsedOn_Options_clist
68 \clist_set:Nn \g__UsedOn_Options_clist { UsedOn, UsedBy, UsedByAndOn }

```

(End of definition for `\g__UsedOn_Options_clist`.)

```

69 \cs_new:Nn \__UsedOn_UsedOnMessage:n { }
70 \cs_new:Nn \__UsedOn_UsedByMessage:n { }
71 \cs_new:Nn \__UsedOn_UsedByAndOnMessage:nn { }
72 \NewDocumentCommand \UsedOnText { m } { \__UsedOn_UsedOnMessage:n { #1 } }
73 \NewDocumentCommand \UsedByText { m } { \__UsedOn_UsedByMessage:n { #1 } }
74 \NewDocumentCommand \UsedByAndOnText { m } { \__UsedOn_UsedByAndOnMessage:n { #1 } }

```

`\SetUsedOnText`

```

75 \NewDocumentCommand \SetUsedOnText { m }
76 {
77   \cs_set:Nn \__UsedOn_UsedOnMessage:n { #1 }
78 }

```

(End of definition for `\SetUsedOnText`. This function is documented on page 7.)

`\SetUsedByText`

```

79 \NewDocumentCommand \SetUsedByText { m }
80 {
81   \cs_set:Nn \__UsedOn_UsedByMessage:n { #1 }
82 }

```

(End of definition for `\SetUsedByText`. This function is documented on page 7.)

`\SetUsedByAndOnText`

```

83 \NewDocumentCommand \SetUsedByAndOnText { m }
84 {
85   \cs_set:Nn \__UsedOn_UsedByAndOnMessage:nn { #1 }
86 }

```

(End of definition for `\SetUsedByAndOnText`. This function is documented on page 7.)

```

87 \RequirePackage{iflang}
88
89 \SetUsedOnText
90 {
91   \IfLanguageName{english}
92     { (Used-on~#1.) } {}
93   \IfLanguageName{french}
94     { (Apparaît-en~#1.) } {}
95   \IfLanguageName{ngerman}
96     { (Wird-auf~#1.) } {}
97   \IfLanguageName{spanish}

```

```

98         { (Aparece~en~#1.) } {}
99     }
100 \SetUsedByText
101 {
102     \IfLanguageName{english}
103     { (Used-by~#1.) } {}
104     \IfLanguageName{french}
105     { (Apparaît~dans~#1.) } {}
106     \IfLanguageName{ngerman}
107     { (Wird-in~#1.) } {}
108     \IfLanguageName{spanish}
109     { (Aparece~en~#1.) } {}
110 }
111 \SetUsedByAndOnText
112 {
113     \IfLanguageName{english}
114     { (Used-by~#1-on~#2.) } {}
115     \IfLanguageName{french}
116     { (Apparaît~dans~#1-en~#2.) } {}
117     \IfLanguageName{ngerman}
118     { (Wird-in~#1-auf~#2.) } {}
119     \IfLanguageName{spanish}
120     { (Aparece~en~#1-en~#2.) } {}
121 }
122

```

\UsedOnMessage The following are the standard texts that get printed in the first line of the labelled environment which later gets referenced with [UsedOn], [UsedBy] or [UsedByAndOn].

```

123 \NewDocumentCommand \UsedOnMessage { m }
124 {
125     \emph{ \_UsedOn_UsedOnMessage:n { #1 } } \[.3\baselineskip]
126 }

```

(End of definition for \UsedOnMessage. This function is documented on page 6.)

\UsedByMessage

```

127 \NewDocumentCommand \UsedByMessage { m }
128 {
129     \emph{ \_UsedOn_UsedByMessage:n { #1 } } \[.3\baselineskip]
130 }

```

(End of definition for \UsedByMessage. This function is documented on page 6.)

\UsedByAndOnMessage

```

131 \NewDocumentCommand \UsedByAndOnMessage { m m }
132 {
133     \emph{ \_UsedOn_UsedByAndOnMessage:nn { #1 } { #2 } } \[.3\baselineskip]
134 }

```

(End of definition for \UsedByAndOnMessage. This function is documented on page 6.)

_UsedOn_Printer Given a $\langle LabelName \rangle$, the following command records all references via the optional **cref** arguments [UsedOn], [UsedBy] or [UsedByAndOn], i.e. when the user called $\backslash\text{cref}[\langle Option \rangle]\{\langle LabelName \rangle\}$. They are recorded in a temporary comma-separated

list (a `clist` in `expl3` speak). This `clist` is then passed to `cleveref`'s `cpageref` or `cref` which in turn is passed to `\UsedOnMessage`, `\UsedByMessage` or `\UsedByAndOnMessage` to be printed after the original label.

```
135 \NewDocumentCommand \__UsedOn_Printer { m m }
136 {
```

First, we will check if the reference $\langle Option \rangle @ \langle LabelName \rangle @ 1$ exists. Here, the `@1` means that $\langle LabelName \rangle$ has been referenced with option $[\langle Option \rangle]$ at least once. If this reference does not exist, nothing happens.

```
137 \cs_if_exist:cT {r@#1@#2@1}
138 {
```

Next, we store all the references of the form $\langle Option \rangle @ \langle LabelName \rangle @ \langle Number \rangle$ in a temporary comma-separated list (`clist`). We do this by looping from 1 to the value of `LastRun@<Option>@<LabelName>` (if the latter value exists, otherwise we set it to 1). Initially, this will need two consecutive runs of `pdflatex`.

```
139 \cs_if_free:cTF {c@LastRun@#1@#2}
140 { \int_set:Nn \l_tmpa_int { 1 } }
141 { \int_set:Nn \l_tmpa_int { \value{LastRun@#1@#2} } }
142 \int_set:Nn \l_tmpb_int { 1 }
143 \int_while_do:nn { \l_tmpb_int <= \l_tmpa_int }
144 {
145 \clist_put_right:Nx \l_tmpa_clist { #1@#2@int_use:N \l_tmpb_int }
146 \int_incr:N \l_tmpb_int
147 }
```

Finally, we print the message that was set in the macro `\UsedOnMessage`.

```
148 \str_case:xn { \str_foldcase:n { #1 } }
149 {
150 {usedon}
151 { \UsedOnMessage { \cpageref{\l_tmpa_clist} } } }
152 {usedby}
153 { \UsedByMessage { \cref{\l_tmpa_clist} } } }
154 {usedbyandon}
155 { \UsedByAndOnMessage { \cref{\l_tmpa_clist} } { \cpageref{\l_tmpa_clist} } } }
156 }
157 }
158 }
```

(End of definition for `__UsedOn_Printer`.)

`__UsedOn_PrintMessage` This macro prints the corresponding `UsedOn` message after the original label.

```
159 \NewDocumentCommand \__UsedOn_PrintMessage { m }
160 {
161 \clist_map_inline:Nn \g__UsedOn_Options_clist
162 {
163 \__UsedOn_Printer{##1}{#1}
164 }
165 }
```

(End of definition for `__UsedOn_PrintMessage`.)

`\l__UsedOn_Option_str` This variable will be used to store the used option in a `\cref` call.

```
166 \str_new:N \l__UsedOn_Option_str
```

(End of definition for \l__UsedOn_Option_str.)

`__UsedOn_Processor` This macro takes an optional argument (a case-insensitive version of the options or their shortform) and a mandatory argument (a single $\langle\textit{LabelName}\rangle$ or a clist $\{\langle\textit{LabelName1}\rangle,\langle\textit{LabelName2}\rangle,\dots\}$).

```

167 \NewDocumentCommand \__UsedOn_Processor { o m }
168   {
169     \IfValueT{#1}{

```

First, we check if the option [UsedOn] or [uo] (case-insensitive) was used.

```

170     \str_case:xnTF { \str_foldcase:n { #1 } }
171     {
172         {usedon}      {\str_set:Nn \l__UsedOn_Option_str {UsedOn}}
173         {uo}          {\str_set:Nn \l__UsedOn_Option_str {UsedOn}}
174         {usedby}      {\str_set:Nn \l__UsedOn_Option_str {UsedBy}}
175         {ub}          {\str_set:Nn \l__UsedOn_Option_str {UsedBy}}
176         {usedbyandon} {\str_set:Nn \l__UsedOn_Option_str {UsedByAndOn}}
177         {ubao}        {\str_set:Nn \l__UsedOn_Option_str {UsedByAndOn}}
178     }
179     {
180     {

```

Loop through the (potential) label list in the mandatory argument of `\cref` (or `\Cref`) which gets passed as the mandatory argument of the current macro.

```

181         \seq_set_from_clist:Nn \l_tmpa_seq {#2}
182         \seq_map_inline:Nn \l_tmpa_seq
183         {

```

If the label has *not* been referenced yet via the option #1 where #1 is one of the current available options in {UsedOn,UsedBy,UsedByAndOn}, create a counter for the current run `ThisRun@<Option>@##1`. If we are not in the initial run anymore, there should be a counter `LastRun@<Option>@##1` which contains the maximal amount this specific label has been referenced via `UsedOn`. If we are in the initial run, we need to create this counter as well. Then save the label in the global container `\g__UsedOn_k_seq`.

```

184         \seq_if_in:NxF \g__UsedOn_k_seq { \l__UsedOn_Option_str @##1 }
185         {
186             \newcounter{ThisRun@ \l__UsedOn_Option_str @##1 }
187             \cs_if_free:cT {c@LastRun@ \l__UsedOn_Option_str @##1}
188             {
189                 \newcounter{LastRun@ \l__UsedOn_Option_str @##1}
190             }
191             \seq_gput_right:Nx \g__UsedOn_k_seq
192             { \l__UsedOn_Option_str @##1}
193         }

```

Increase the counter for the current run by 1 and set the counter for last run (containing the maximal amount of `UsedOn-\cref`'s) to...the maximal amount of `UsedOn-\cref`'s.

```

194         \stepcounter{ThisRun@ \l__UsedOn_Option_str @##1}
195         \setcounter{LastRun@ \l__UsedOn_Option_str @##1}
196         {
197             \fp_eval:n
198             {
199                 max(
200                     \value{ThisRun@ \l__UsedOn_Option_str @##1} ,

```

```

201         \value{LastRun@ \l__UsedOn_option_str @##1}
202     )
203 }
204 }

```

Store the value of the max counter LastRun@<Option>@##1 in the global container \g__UsedOn_kv_prop.

```

205 \prop_gput:Nxx \g__UsedOn_kv_prop
206 { \l__UsedOn_option_str @##1}
207 {\arabic{LastRun@ \l__UsedOn_option_str @##1}}

```

Now we create a numbered auxiliary label. This label is issued at the location where we referenced the original label via \cref[UsedOn]<LabelName>. The new auxiliary label has the prefix UsedOn@, UsedBy@ or UsedByAndOn@ and the suffix @\arabic{ThisRun@<Option>@##1}, e.g. UsedOn@thm:Pythagoras@4 if it is the fourth time that we called

\cref[UsedOn]{thm:Pythagoras}.

```

208         \__UsedOn_origlabel
209     {
210         \l__UsedOn_option_str @##1@
211         \arabic{ThisRun@ \l__UsedOn_option_str @##1}
212     }
213 }
214 }
215 }
216 {

```

Throw an error, if an unrecognised option was used for the optional argument to this macro.

```

217 \msg_new:nnn {cleveref-usedon} { OptionSpellingError }
218 {
219     \MessageBreak
220     Spelling-error~\msg_line_context:
221     \MessageBreak
222     Did-you-mean-to-pass-option~\MessageBreak
223     'UsedOn'~to~cref~or~Cref?
224 }
225 \msg_fatal:nn { cleveref-usedon } { OptionSpellingError }
226 }
227 }
228 }

```

(End of definition for __UsedOn_Processor.)

__UsedOn_cref This is just a wrapper around cleveref's \cref. Additionally the __UsedOn_Processor gets called.

```

229 \bool_new:N \l__UsedOn_already_specified_bool
230 \clist_new:N \l__UsedOn_args_clist
231 \cs_new:Nn \__UsedOn_define_from_orig:NN
232 {
233     \NewDocumentCommand #1 { s o m }
234     {
235         \IfValueTF { ##2 }
236         {
237             \bool_set_false:N \l__UsedOn_already_specified_bool

```

```

238 \clist_set:Nn \l__UsedOn_args_clist { ##2 }
239 \clist_put_right:No \l__UsedOn_args_clist { \l__UsedOn_default_tl }
240 \clist_map_inline:Nn \l__UsedOn_args_clist
241 {
242   \str_case:xnT { \str_foldcase:n { #####1 } }
243   {
244     {usedon} {}
245     {uo} {}
246     {usedby} {}
247     {ub} {}
248     {usedbyandon} {}
249     {ubao} {}
250   }
251   {
252     \clist_remove_all:Nn \l__UsedOn_args_clist { #####1 }
253     \bool_if:NF \l__UsedOn_already_specified_bool
254     {
255       \__UsedOn_Processor[#####1]{##3}
256     }
257     \bool_set_true:N \l__UsedOn_already_specified_bool
258   }
259 }
260 \clist_if_empty:NTF \l__UsedOn_args_clist
261 {
262   \IfBooleanTF{##1}{ #2*{##3} }{ #2{##3} }
263 }
264 {
265   \IfBooleanTF{##1}{ #2*[ \l__UsedOn_args_clist ]{##3} }{ #2[ \l__UsedOn_args_clist ]{##3} }
266 }
267 }
268 {
269   \tl_if_blank:VF \l__UsedOn_default_tl
270   {
271     \clist_clear:N \l__UsedOn_args_clist
272     \clist_put_right:No \l__UsedOn_args_clist { \l__UsedOn_default_tl }
273     \clist_map_inline:Nn \l__UsedOn_args_clist
274     {
275       \__UsedOn_Processor[#####1]{##3}
276     }
277   }
278   \IfBooleanTF{##1}{ #2*{##3} }{ #2{##3} }
279 }
280 }
281 }
282 \__UsedOn_define_from_orig:NN \__UsedOn_cref \__UsedOn_origcref
283 \__UsedOn_define_from_orig:NN \__UsedOn_Cref \__UsedOn_origCref
284 \__UsedOn_define_from_orig:NN \__UsedOn_labelcref \__UsedOn_origlabelcref

```

(End of definition for __UsedOn_cref.)

__UsedOn_ReadFromAux From the .aux file we will read the contents of the global container \g__UsedOn_kv_prop. This is a key-value property list and we create and set a for each label (key) and the maximal amount (value) it was called in the last run.

```

285 \NewDocumentCommand \__UsedOn_ReadFromAux { }

```

```

286 {
287   \prop_map_inline:Nn \g__UsedOn_kv_prop
288   {
289     \newcounter{LastRun@##1}
290     \setcounter{LastRun@##1}{##2}
291   }
292 }

```

(End of definition for __UsedOn_ReadFromAux.)

__UsedOn_WriteToAux For each label we write a line in the .aux file of the form:

$\langle \text{LabelName} \rangle = \langle \text{Maximal references via UsedOn in last run} \rangle$.

This information can be constructed from the global container \g__UsedOn_k_seq and the counters with prefix ThisRun@ we set earlier. We need to wrap this in the on/off switch for expl3 functionality.

```

293 \NewDocumentCommand \__UsedOn_WriteToAux { }
294 {

```

First, we clear the global key-value prop list \g__UsedOn_kv_prop and then we rebuild it with the information from the current run.

```

295   \prop_clear:N \g__UsedOn_kv_prop
296   \seq_map_inline:Nn \g__UsedOn_k_seq
297   { \prop_gput:Nxx \g__UsedOn_kv_prop {##1}{\arabic{ThisRun@##1}} }
298   \iow_now:cx { @auxout }
299   { \token_to_str:N \ExplSyntaxOn }

```

Loop through the key-val proplist and write contents to .aux file.

```

300   \prop_map_inline:Nn \g__UsedOn_kv_prop
301   {
302     \tl_set:Nn \l_tmpa_tl { ##1 }
303     \tl_replace_all:Nnn \l_tmpa_tl { ~ } { \c_tilde_str }
304     \iow_now:cx { @auxout }
305     {
306       \prop_gput_from_keyval:Nn \token_to_str:N \g__UsedOn_kv_prop
307       { {\l_tmpa_tl} = ##2 }
308     }
309   }
310   \iow_now:cx { @auxout }
311   { \token_to_str:N \ExplSyntaxOff }
312 }%

```

(End of definition for __UsedOn_WriteToAux.)

At the hook \AtBeginDocument we read from the .aux file and patch commands.

```

313 \hook_gput_code:nnn { begindocument } { cleveref-usedon }
314 {
315   \__UsedOn_ReadFromAux

```

Patch label and cref to include the new [UsedOn] capabilities.

```

316   \NewCommandCopy \__UsedOn_origlabel \label
317   \NewCommandCopy \__UsedOn_origcref \cref
318   \NewCommandCopy \__UsedOn_origCref \Cref
319   \NewCommandCopy \__UsedOn_origlabelcref \labelcref
320   \RenewDocumentCommand \label { m }
321   {

```



```

322     \_UsedOn_origlabel{#1}\_UsedOn_PrintMessage{#1}
323   }
324   \RenewCommandCopy \cref \_UsedOn_cref
325   \RenewCommandCopy \Cref \_UsedOn_Cref
326   \RenewCommandCopy \labelcref \_UsedOn_labelcref
327 }
    At the hook \AtEndDocument we write to the .aux file.
328 \hook_gput_code:nnn { enddocument } { cleveref-usedon }
329 {
330   \_UsedOn_WriteToAux
331 }
332 \endpackage

```