

cdcmd, 一跟可扩展的条件化命令宏包

雾月, Longaster*

2021 年 10 月 12 日 v1.0

cdcmd 提供了条件化命令的方式, 与 styledcmd 宏包不同, 该宏包的某些命令是可 f-扩展的, 这在某些情况下是有用的。

第 1 节 主要接口

<code>\newcondition</code>	<code>\newcondition <{identifier}> <{ids}></code>
<code>\setcondition</code>	<code>\setcondition + <{identifier=ids list}></code>
<code>\clearcondition</code>	<code>\clearcondition[<{identifiers}>]</code>

`\newcondition` 定义 `<identifier>` 的 `<ids>`。`<identifier>` 两侧的空格将被移除。

`\setcondition` 设置在当前组中有效的 `<id>`。不带 + 的版本将局部地清除原先设置的 `<identifier>` 的 `ids`。

`<identifier>`、`<id>` 均不能为 *。

`\clearcondition` 局部地清除指定的 `<identifiers>`, 默认值为 *, 即清除所有。

<code>\conditionif</code>	<code>\conditionif * [<{identifier=ids list}>] <{true}> <{false}></code>
<code>\conditioncmd</code>	<code>\conditioncmd * [<{identifier=ids list}>] <{material}></code>
<code>\econditionif</code>	当 <code><{identifier=ids list}></code> 的组合使得当前条件为真时, 使用 <code><true></code> 、 <code><material></code> , 为假时, 使用
<code>\econditioncmd</code>	<code><false></code> 。

不带 * 的版本为 any, 带 * 的为 all。详见下方的说明。

`\econditionif`、`\econditioncmd` 为可 f-扩展的。`\conditionif`、`\conditioncmd` 为不可扩展(`\protected`)的。

`<{identifier=ids list}>` 默认值为 *, 即在默认情况下将输出 `<true>`、`<material>`。

<code>\conditioncase</code>	<code>\conditioncaseTF * !</code>
<code>\conditioncaseTF</code>	{
<code>\econditioncase</code>	<code><{identifier-ids list case₁>} <{code₁>}</code>
<code>\econditioncaseTF</code>	<code><{identifier-ids list case₂>} <{code₂>}</code>
	...
	<code><{identifier-ids list case_n>} <{code_n>}</code>
	}
	<code><{true code}></code>
	<code><{false code}></code>

! 未给出时, 依次计算 `<{identifier-ids list cases}>`, 直到遇到第一个 `<case>` 为真, 输出对应的 `<code>` 与 `<true code>`, 若未成功匹配, 则输出 `<false code>`。

! 给出时, 依次计算 `<{identifier-ids list cases}>`, 直到遇到第一个 `<case>` 为假, 输出对应的 `<code>` 与 `<true code>`, 若未成功匹配, 则输出 `<false code>`。

T_EXhackers note: 对 `<ids>` 的处理使用的是 L^AT_EX3 的 `\clist_map...`, 它们将形如 `{,}` 的 list 视为空, 形如 `{{,}}` 视为非空。

假定已经使用了如下命令:

*Email: longaster@163.com

```
\newcondition{defined}{}
\newcondition{paper}{a4,a5,b5}
\setcondition{paper}={a5,b5}
```

即 `defined` 已被 `\newcondition` 定义且无 `id`, `paper` 已被 `\newcondition` 定义且有 `a4,b5,a5` 三个 `id`, 其中 `a5,b5` 已被设置, `undefined` 未被定义。

`any` 在满足如下条件之一为真:

1. $\langle identifier=ids list \rangle$ 为 `*`;
2. $\langle identifier=ids list \rangle$ 为一个已被定义的 $\langle identifier \rangle$, 且未给出 $\langle id \rangle$ 值, 即 $\langle identifier=ids list \rangle$ 为 `paper` 或 `defined`;
3. $\langle identifier=ids list \rangle$ 为一个已被定义的 $\langle identifier \rangle$, 且 $\langle id \rangle$ 值为 `*`, 即 $\langle identifier=ids list \rangle$ 为 `paper=*` 或 `defined=*`;
4. $\langle identifier=ids list \rangle$ 为一个已被定义的 $\langle identifier \rangle$, 且给出的 $\langle ids \rangle$ 中有被设置的 `id`, 即 $\langle identifier=ids list \rangle$ 为 `paper=b5` 或 `paper={a5,b5}` 或 `paper={a5,a0}` (`a0` 未设置, 但 `a5` 已设置)。注意 `defined` 给出的任何除 `*` 之外的 $\langle ids \rangle$ 值都将使得条件为假, 即使是空值 (`defined=`);
5. $\langle identifier=ids list \rangle$ 中任意一项满足如上 4 点之一, 如 `paper={a5,a0},undefined`。

`all` 在满足如下条件之一为真:

1. $\langle identifier=ids list \rangle$ 为 `*`;
2. $\langle identifier=ids list \rangle$ 为一个已被定义的 $\langle identifier \rangle$, 且未给出 $\langle id \rangle$ 值, 即 $\langle identifier=ids list \rangle$ 为 `paper` 或 `defined`;
3. $\langle identifier=ids list \rangle$ 为一个已被定义的 $\langle identifier \rangle$, 且 $\langle id \rangle$ 值为 `*`, 即 $\langle identifier=ids list \rangle$ 为 `paper=*` 或 `defined=*`;
4. $\langle identifier=ids list \rangle$ 为一个已被定义的 $\langle identifier \rangle$, 且给出的 $\langle ids \rangle$ 均被设置, 即 $\langle identifier=ids list \rangle$ 为 `paper=b5` 或 `paper={a5,b5}`。注意 `defined` 给出的任何除 `*` 之外的 $\langle id \rangle$ 值都将使得条件为假, 即使是空值 (`defined=`);
5. $\langle identifier=ids list \rangle$ 中所有项均满足如上 4 点之一, 如 `paper={a5,b5},defined`。

```
\newconditioncommand
\renewconditioncommand
\provideconditioncommand
\declareconditioncommand
\neweconditioncommand
\reneweconditioncommand
\provideweconditioncommand
\declareeconditioncommand
```

```
\newconditioncommand * <function> [<arg nums>] [<default>] {<code>}
\neweconditioncommand * <function> [<arg nums>] {<code>}
```

这些命令与 $\text{\LaTeX} 2_{\epsilon}$ 的命令相对应。`\newconditioncommand` 命令正如 $\text{\LaTeX} 2_{\epsilon}$ 的 `\newcommand` 命令。它们将定义形如 `\foo+{<identifier=ids list>}<args>` 的命令。定义的命令的可选参数不可包含 `\par`。

`\neweconditioncommand` 命令定义的是可扩展的命令, 不可设置默认参数。但可使用 `xparse` 形式的命令定义可设置默认值的参数。

不带 `*` 的版本定义的命令是 `\long` 的。即其参数可以包含多个段落。

```
\NewConditionCommand          \NewConditionCommand <function> {<arg spec>} {<code>}
\RenewConditionCommand
\ProvideConditionCommand
\DeclareConditionCommand
\NewExpandableConditionCommand
\RenewExpandableConditionCommand
\ProvideExpandableConditionCommand
\DeclareExpandableConditionCommand
```

这些命令与 `xparse` 宏包的命令相对应。它们将定义形如 `\foo+{<identifier=ids list>}<args>` 的命令。

$\langle arg spec \rangle$ 应符合 `xparse` 宏包相应命令的规则。

其中带 `+` 的命令为 `all`, 不带 `+` 的为 `any`。 $\langle identifier=ids list \rangle$ 的含义如上所述。 $\langle args \rangle$ 为由 $\langle arg spec \rangle$ 定义的参数。

第2节 例子

```

\newcondition{defined}{}
\newcondition{paper}{a4,a5,b5}
\setcondition{paper}={a5,b5}

\conditionif [*]{t}{f}: t
\conditionif [defined]{t}{f}: t
\conditionif [defined=]{t}{f}: f
\conditionif [defined=*]{t}{f}: t
\conditionif [defined=a]{t}{f}: f
\conditionif [paper={a5,a0},undefined]{t}{f}: t

\conditionif [*]{t}{f}: t
\conditionif *[defined]{t}{f}: t
\conditionif *[defined={,,}]{t}{f}: f
\conditionif *[defined=*]{t}{f}: t
\conditionif *[defined=a]{t}{f}: f
\conditionif *[paper={a5,a0},undefined]{t}{f}: f
\conditionif *[,undefined]{t}{f}: f
\conditionif *[paper={a5,b5}]{t}{f}: t
\conditionif *[paper={a5,,b5}]{t}{f}: t
\conditionif *[paper={a5,b6,a5}]{t}{f}: f
\conditionif *[paper={a5,{ },45}]{t}{f}: f
\conditionif *[,defined,paper={a5,b5}]{t}{f}: t

\def\truetext{true} \def\falsetext{false}
\edef\testa{\econditionif[*]{true}{false}}
\ifx\testa\truetext t\else f\fi
\ifx\testa\falsetext t\else f\fi
\strcmp {\econditionif[*]{true}{false}} {true}
\strcmp {\econditionif[*]{true}{false}} {false}
\strcmp {\testa} {\truetext}
\strcmp {\testa} {\falsetext}

tf0 1 0 1

\conditioncase{
  {paper=a3} {a3}
  {paper=a4} {a4}
  {paper,defined} {pd}
}

pd

\conditioncaseTF{
  {paper=a3} {a3}
  {paper=a4} {a4}
  {paper,defined} {pd}
}{true}{false}

pdtrue

\conditioncase!{
  {paper=a3} {a3}
  {paper=a4} {a4}
  {paper,defined} {pd}
}

a3

```

```

\conditioncaseTF!{
  {paper=a3} {a3}
  {paper=a4} {a4}
  {paper,defined} {pd}
}{true}{false}

a3true

\newconditioncommand\longprotectedcdcmd{longprotectedcdcmd}
\newconditioncommand\longprotectedcdcmdi[1]{longprotectedcdcmdi<#1>}
\newconditioncommand\longprotectedcdcmdio[1][DFT]{longprotectedcdcmdio<#1>}
\newconditioncommand*\shortprotectedcdcmd{shortprotectedcdcmd}
\newconditioncommand*\shortprotectedcdcmdi[1]{shortprotectedcdcmdi<#1>}
\newconditioncommand*\shortprotectedcdcmdio[1][DFT]{shortprotectedcdcmdio<#1>}

\setcondition{paper={a4,a5}}
\longprotectedcdcmd{*}
\longprotectedcdcmdi{*}{1\par arg}
\longprotectedcdcmdio{*}
\longprotectedcdcmdio{*}[1opt]
\longprotectedcdcmdio{paper=a4}[1opt a4]
\longprotectedcdcmdio+{paper={a4,a7}}[1opt a4a7]
\shortprotectedcdcmd{*}
\shortprotectedcdcmdi{*}{1\par arg}
\shortprotectedcdcmdio{*}
\shortprotectedcdcmdio{*}[1opt]
\shortprotectedcdcmdio{paper=a4}[1opt a4]
\shortprotectedcdcmdio+{paper={a4,a7}}[1opt a4a7]

longprotectedcdcmd
longprotectedcdcmdi<1
arg>
longprotectedcdcmdio<DFT>
longprotectedcdcmdio<1opt>
longprotectedcdcmdio<1opt a4>
shortprotectedcdcmd
shortprotectedcdcmdi<1arg>
shortprotectedcdcmdio<DFT>
shortprotectedcdcmdio<1opt>
shortprotectedcdcmdio<1opt a4>

```

第 3 节 For package author

```

\cdcmd_any_if_p:n *
\cdcmd_any_if_p:(o|V|f) *
\cdcmd_any_if:nTF *
\cdcmd_any_if:(o|V|f)TF *
\cdcmd_all_if_p:n *
\cdcmd_all_if_p:(o|V|f) *
\cdcmd_all_if:nTF *
\cdcmd_all_if:(o|V|f)TF *

```

它们的含义应该是显然的。

```

\cdcmd_any_case_true:nTF
\cdcmd_any_case_false:nTF
\cdcmd_all_case_true:nTF
\cdcmd_all_case_false:nTF

```

它们的含义应该是显然的。

第 4 节 实现

见 [cdcmd.pdf](#)