

I. Use cases in L^AT_EX

Documenting a package, or class, is best done through use cases showing both the code and the corresponding result.¹

1. “Inline” codes

Example I.1 (Standard use). The `\tdoclatexin` macro² can be used to type code in line in a similar way to `\verb`, or as a standard macro (see the handling of braces in the latter case below). Here are some examples of use.³

1: <code>\tdoclatexin/\$a^b = c\$</code>	1: $a^b = c$
2: <code>\tdoclatexin+\tdoclatexin/\$a^b = c\$/+</code>	2: <code>\tdoclatexin/\$a^b = c\$</code>
3: <code>\tdoclatexin{\tdoclatexin{\$a^b = c\$}}</code>	3: <code>\tdoclatexin{\$a^b = c\$}</code>

Example I.2 (Possible options). As the `\tdoclatexin` macro is based on `minted`, you can use all the options taken into account by `minted`. Here are some examples.

1: <code>\tdoclatexin[style = bw]{\$a^b = c\$}</code>	1: $a^b = c$
2: <code>\tdoclatexin[style = igor, showspaces]{\$a^b = c\$}</code>	2: $a^b = c$

Note.

The `\tdoclatexin` macro can be used in a footnote as shown below.^a

^a`$minted = TOP$` has been typed `\tdoclatexin+$minted = TOP$+` in this footnote.

2. Directly typed codes

Example I.3 (Side by side). Displaying a code and its rendering side by side is done as follows where the macro `\tdoctcb` allows you to just type `\tdoctcb{sbs}` instead of `listing side text` (`sbs` is for “s·ide b·y s·ide”, while `tcb` is the standard abbreviation for `tcolorbox`). Note the use of rafters, not square brackets (more on this later).

<code>\begin{tdoctx}<\tdoctcb{sbs}></code>
$A = B + C$
<code>\end{tdoctx}</code>

This gives :

$A = B + C$	$A = B + C$
-------------	-------------

Example I.4 (Following). `\begin{tdoctx}... \end{tdoctx}` produces the following result (this default setting is also obtained by using `\tdoctcb{std}`).⁴

$A = B + C$
$A = B + C$

Example I.5 (Just the code). Via `\tdoctcb{code}`, we’ll just get the code as below.

$A = B + C$

Example I.6 (Customise). The `tdoctx` environment accepts two types of optional argument.

1. Between classic square brackets, you can use any option taken into account by `minted`.
2. Between rafters, you can use any option managed by the environments obtained via `tcolorbox`.

For example, the following modifications can be made if required.⁵

¹Code is formatted using the `minted` and `tcolorbox` packages.

²The name of the macro `\tdoclatexin` comes from “in·line L^AT_EX”.

³A background color is deliberately used to subtly highlight the L^AT_EX codes.

⁴`std` refers to the “standard” behaviour of `tcolorbox` in relation to the `minted` library.

⁵This documentation uses the options between rafters to obtain correct rendering of code producing shaded frames: see the section ?? on page ??.

```

\begin{tdoclatex}%
  [linenos, style = igor, showspace]%
  <\tdoctcb{sbs},
  attach boxed title to top left = {yshift = -9pt},
  fonttitle                       = \bfseries,
  title                           = Local modifications,
  top                             = 10pt>
% Sometimes useful, but don't overuse it!
$A = B + C$
% End of this demonstration.
\end{tdoclatex}

```

This gives :

Local modifications

```

1 %_Sometimes_useful,_but_don't_overuse_it!
2 $A=_B_+_C$
3 %_End_of_this_demonstration.

```

$A = B + C$

⚠ Warning.

To obtain the default formatting for a code beginning with a bracket or a rafter, you'll need to do a bit of fiddling, as shown below.

```

\begin{tdoclatex}[]
[Strange... Or not!]
\end{tdoclatex}
OR.
\begin{tdoclatex}<>
<Strange... Or not!>
\end{tdoclatex}

```

This gives :

```

[Strange... Or not!]
-----
[/Strange... Or not!/]

```

OR.

```

<Strange... Or not!>
-----
</Strange... Or not!>

```

Another method is to use the `\string` primitive, as shown below.

```

\begin{tdoclatex}
\string[Strange... Or not!]
\end{tdoclatex}
OR.
\begin{tdoclatex}
\string<Strange... Or not!>
\end{tdoclatex}

```

This gives :

```

[Strange... Or not!]
-----
[/Strange... Or not!/]

```

OR.

```

<Strange... Or not!>
-----
</Strange... Or not!>

```