# loop.tex

A simple looping construct (meta-macros).
Version: 1.0 (15 April 1991) (doc: 15 Apr 1991)


The macros in loop.tex are intended to help you write your own macros, rather than to typeset text directly. Consequently, it is assumed that you have some experience writing macros; In other words, assume that this whole section is preceded by a "dangerous bend" symbol. These macros are kept simple and are not very robust: if you decide to use them in a different way than described below you are entirely responsible for what happens[1].

Loop.tex defines an alternative to plain TeX's `\loop` construct. (It can also be used with, e.g., LaTeX.) Plain TeX's `\loop` is often awkward to use because of the limited form of tests in TeX. The new construct tries to alleviate this problem by allowing a more flexible use of tests in the loop. To execute a loop, you say `\Loop ... \Pool`[2]. This will normally execute the loop 'body', i.e., the commands corresponding to the dots, infinitely many times. There are two special commands you can use within the loop, `\Break` and `\Continue`. When a `\Break` command is executed, the rest of the body is ignored, and the loop is terminated. Execution of a `\Continue` command also causes the rest of the body to be ignored, but then the loop is started again from the beginning.

Both `\Break` and `\Continue` must be followed by a number. This number should be the number of `\if...` constructs that are terminated by the command. In other words, it is the number of `\fi`'s that has to be 'executed' to get proper balancing of all `\if...` tests. It is somewhat unfortunate that this number has to be supplied, but there seems to be no way to close all `\if...` tests automatically. Here is an example similar to one from the TeXbook.

```
\def\yes{Yes } \def\no{No }
\Loop\message{Are you happy? }
     \read-1 to\answer
     \ifx\answer\yes \Break1
     \else\ifx\answer\no \Continue2
     \fi  \fi
     \message{(Please type Yes or No.)}
\Pool
```

At the position of `\Break` the nesting level of `\if...` tests is 1, at the position of `\Continue` the level is 2. If you forget the number you will (usually) get a '`missing number`' error message, but possibly at an unexpected point in the program. The wrong number typically results in either an '`Extra \fi`' or an '`\if... was incomplete`' message.

There can be any number of `\Break` and `\Continue` commands in a loop, although it is wise to have at least one `\Break` command. The text following any of these commands up till the `\Pool` at the end of the loop must have balanced braces, and the loop body itself must also have balanced braces. If this causes a problem, you can use `\bgroup` and `\egroup` instead of `{` and `}`. If you want to nest two loops, you will have to enclose the inner loop within a group. (Incidentally, these restrictions on balancing braces and nesting loops also apply to plain TeX's `\loop` construct.)

---

[1] Not that the author assumes any responsibility even if you *do* obey the rules.

[2] We use initial capital letters to prevent confusion with plain TeX's `\loop`. Although you don't need `\loop` anymore, there may still be macros that depend on it (e.g., plain TeX's tabbing macros).