The `padcount` Package, v1.0

Donald P. Goodman III

May 5, 2018

Abstract

Sometimes a plain number is fine; other times, you want it *padded*, either with zeroes or some other character. This small package makes it easy to do that. It works not only on arabic numerals, but on any expanded list of tokens passed to it. This makes it suitable for, among other things, counters of all kinds.

1 Usage

`\padnum` About as simply as possible, the package centers on the `\padnum` macro, which is defined as follows:

`\padnum {`⟨*stuff*⟩`}`

The argument can be anything that TeX will recognize as a list of tokens, though it's anticipated that it'll be used mostly for numbers. By default, `padcount` pads to two digits, and it does that padding with the character `0`.

The examples in this document assume the existence of a TeX count register `\tmp` (created by `\newcount\tmp`) and a LaTeX counter `temp` (created by `\newcounter{temp}`).

`\padnum{4}`	04	`\padnum{47}`	47
`\tmp=8\padnum{\the\tmp}`	08	`\tmp=16\padnum{\the\tmp}`	16

`\setcounter{temp}{8}`
`\padnum{\thetemp}`
	08	`\setcounter{temp}{16}`
		`\padnum{\thetemp}`
			16

`\setpadnum` You can change the number of digits you'd like to pad to with the macro `\setpadnum`, which takes a single integer as its argument. By default, this is 2; but any integer whatsoever will work.

`\setpadnum{4}\padnum{4}`	0004
`\setpadnum{8}\padnum{42424}`	00042424

`\setpadchar` You can set the character that will be used for padding with `\setpadchar`. In fact, this need not be a character; it can be any expansible set of tokens.

`\setpadnum{4}\setpadchar{*}=\padnum{4}=` =***4=

`\setpadnum{4}\setpadchar{\hskip1.2ex}=\padnum{4}=` =   4=

`\setpadnum{4}\setpadchar{*}=\padnum{\roman{temp}}=` =*xvi=

That last example, incidentally, also demonstrates that we can send a counter in any format, without regard to base, and `padcount` will do its job.

And that's it; just a small utility package. Hope it's useful in some way.

2 Implementation

We start with the macros to count characters, which we've gently adapted from those of David Carlisle's answer on `tex.stackexchange.com`, question 57598.

```
\newcount\pad@charcount%
\def\pad@expandloop#1{%
\pad@xloop#1\relax
}
\def\pad@xloop#1{%
\ifx\relax#1%
\else%
\advance\pad@charcount by1%
\expandafter\pad@xloop%
\fi%
}%
\def\pad@countchar#1{%
\pad@charcount=0%
\expandafter\expandafter\pad@expandloop{#1}%
}%
```

Now, we can set the defaults. Set the padding level to two characters and the padding character to `0`. Also define some throwaway counters for loops and value-holding.

```
\newcount\pad@tmp%
\newcount\pad@tmpi%
\newcount\pad@num\pad@num=2%
\def\pad@char{0}%
```

Now, the money: count the characters, determine how many new characters we need to print, then loop the appropriate number of times to print them.

```
\def\padnum#1{%
\edef\pad@yy{#1}%
\expandafter\pad@countchar{\pad@yy}%
\pad@tmp=\pad@charcount%
\multiply\pad@tmp by-1%
\advance\pad@tmp by\pad@num%
\pad@tmpi=0%
\ifnum\pad@tmp>0%
\loop%
\ifnum\pad@tmpi<\pad@tmp%
\pad@char%
\advance\pad@tmpi by1%
\repeat%
\fi%
\leavevmode#1%
}%
```

Lastly, some macros to reset the internal values used by `\padnum`.

```
\def\setpadnum#1{\pad@num=#1}%
\def\setpadchar#1{\def\pad@char{#1}}%
```

Happy TeXing!