stricttex – strictly balanced brackets and numbers in command names (v0.2β)

Sebastian Ørsted (sorsted@gmail.com)

September 8, 2020

The stricttex package is a small, LuaLaTeX-only package providing you with three, sometimes useful features:

- It allows you to make brackets `[...]` "strict", meaning that each `[` must be balanced by a `]`.
- It allows you to use numbers in command names, so that you can do stuff like `\newcommand\pi12{\pi_{12}}`.
- It allows you to use numbers *and* primes in command names, so that you can do stuff like `\newcommand\pi'12{\pi '_{12}}`.

Making brackets strict

The package provides the commands

```
\StrictBrackets0n
\StrictBrackets0ff
```

Between these two commands, all left brackets `[` are replaced by `[{`, and all right brackets `]` by `}]`. This forces the brackets to be properly balanced. This is extremely useful in come packages, such as SemanTeX, where you can then do things that would otherwise cause errors, e.g.

```
\StrictBrackets0n
  $ \vf[upper=\vx[upper=2,lower=3]] $
\StrictBrackets0ff
```

Normal brackets can still be accessed by using the standard TeX commands `\lbrack` and `\rbrack`. The replacement algorithm has two important exceptions:

- *No* replacements apply to the commands `\[...\]`, which can therefore be used as normal.
- If you absolute *need* ordinary brackets, you can write `<[>` and `<]>` to access them. This works in all contexts, so e.g. `\<[>` and `\<]>` will work just like `\[` and `\]`.

Allowing numbers (and possibly primes) in commands

The package provides the commands

```
\NumbersInCommandsOn
\NumbersInCommandsOff
\NumbersAndPrimesInCommandsOn
\NumbersAndPrimesInCommandsOff
```

The first pair of commands allows you to define commands containing numbers. So the following will work:

```
\NumbersInCommandsOn
\newcommand\pi12{\pi_{12}}
\newcommand\pi13{\pi_{13}}
\newcommand\pi23{\pi_{23}}
\newcommand\pi12comma34{\pi_{12,34}}
\NumbersInCommandsOff
```

Internally, what happens is that if a command is immediately followed by a number, that numbers is replaced by a text string, i.e. `0` gets replaced by `numberZERO`, `1` gets replaced by `numberONE`, etc. These long names have been chosen to prevent name clashes. In other words, the code that is eventually passed to TEX is

```
\newcommand\pinumberONEnumberTWO{\pi_{12}}
\newcommand\pinumberONEnumberTHREE{\pi_{13}}
\newcommand\pinumberTWOnumberTHREE{\pi_{23}}
\newcommand\pinumberONEnumberTWOcommanumberTHREEnumberFOUR{\pi
_{12,34}}
```

Needless to say, stuff like `\kern11pt` will no longer work and will have to be replaced by `\kern 11pt`.

The commands `\NumbersAndPrimesInCommandsOn` and `\...Off` work almost the same way, except they also allow you to use *primes*. So the following will work:

```
\NumbersAndPrimesInCommandsOn
\newcommand\pi'12{\pi '_{12}}
\NumbersAndPrimesInCommandsOff
```

Internally, the algorithm works as before, except the prime `'` gets replaced by `symbolPRIME`. So what is eventually passed to TEX is

```
\newcommand\pisymbolPRIMEnumberONEnumberTWO{\pi '_{12}}
```