siunitx: Launching version 3

Joseph Wright

1 Introduction

Typesetting units is an important task for scientists and engineers. When done in (LA)TEX, it is natural to use a macro-based approach to carry out the formatting. I entered this area in 2008 with the LATEX package siunitx, which I've looked at in *TUGboat* before (Wright, 2011; Wright, 2018). Last year, I launched version 3 of siunitx after development over several years. Here, I will recap why this came about and how I went about launching the latest version.

2 Looking back

I started developing siunitx in late 2007, when I answered a question about a bug in a previous package, SIunits (Heldoorn and Wright, 2007). I quickly decided to write a new package, which combined SIunits and SIstyle (Els, 2008) with a key–value interface and bringing in some new ideas. The first release of siunitx was available at Easter 2008.

This first release worked well, and soon picked up lots of users. However, internally it was essentially a mix of material from the previous packages put together. It also had some poor choices for key names, which I very much wanted to address. I therefore wanted to work on a second implementation, and got on to that in 2010.

Around this time, I started working with expl3 coding, and quickly decided to move siunitx to using expl3 internally. The re-write meant starting from scratch in some ways, but I got a lot of work done pretty quickly and was quite happy with the results.

3 Moving forward again

Version two retains most of the features that version one had, but as well as the good ones, it turns out it retained some issues, particular in the internal API. I needed to look at a system where the different parts of the siunitx system communicated with each other using a documented approach. To support that, and to keep things working with existing sources, I needed tests. And there was a big issue with fonts.

The font assumptions in versions 1 and 2 carry all the way through from SIstyle (Els, 2008), and which I adjusted only slightly. The approach was:

1. Detect the current font type using LATEX internal data.
2. Insert everything inside `\text` (an `\mbox`).
3. Apply `\ensuremath` inside the box.
4. Perhaps apply `\text` again (for text mode output).
5. Force the font by using *e.g.* `\mathrm` or `\rmfamily`.

That requires a lot of work, and it's not always easy to get it right. So there is a new approach for version three:

1. Detect current font type using LATEX internal data.
2. Set any aspects *that are needed*.
3. Only use an `\mbox` if math version has to be altered.

This 'minimal change' approach is much faster than the current one, and is much better at respecting font changes. The downside is that this is quite hard to map in all cases to the older keys: we were going to need a way to deal with this.

3.1 Making it all work

Getting version 3 out needed me to solve three major issues:

1. Testing;
2. Backward compatibility;
3. Handling multi-part and complex values.

Only one of those (the third) is an area for *just me*: the first two are general, and tools from the LATEX team have helped. Over the past few years, we have developed l3build (Wright, 2022) as an automated testing tool. So development of version 3 has been backed by testing for *everything*: each code-level API and all of the key–value interfaces have dedicated tests.

It's not possible to be entirely backward-compatible with the scale of the changes I've made in siunitx. Luckily, the LATEX team have also provided a mechanism to handle this. The user can add

```
\usepackage{siunitx}[=v2]
```

and this will load the last version of `v2`; all I have to do is set things up as a package author. The siunitx source thus contains:

```
% Set up a couple of commands in recent-ish
% \LaTeXe{} releases.
\providecommand\DeclareRelease[3]{}
\providecommand\DeclareCurrentRelease[2]{}
%
% Allow rollback to version~$2$: if we need to,
% version~$1$ could eventually be added here too.
\DeclareRelease{2}{2010-05-23}{siunitx-v2.sty}
\DeclareRelease{v2}{2010-05-23}{siunitx-v2.sty}
\DeclareCurrentRelease{}{2021-05-17}
```

The third issue above was dealing with multi-part quantities (*e.g.* $2\,\mathrm{m} \times 3\,\mathrm{m}$) and complex numbers ($1 + 2\mathrm{i}$). I decided to keep the core code faster, and to provide dedicated document commands for these.

Again, this means you might have to update a source to go from version 2 to 3, but I think this is the right call.

4 Conclusions

It's taken a few years, but with the tools available from the LaTeX team, creating a new version of siunitx has worked well. Over a year after the launch, the code is performing well and I've dealt with the minor issues that came up. I'm now looking forward to developing new features that have been outstanding for years.

References

Els, D. "The SIstyle package". 2008. ctan.org/pkg/sistyle.

Heldoorn, Marcel, and J. Wright. "The SIunits package: Consistent application of SI units". 2007. ctan.org/pkg/siunits.

Wright, Joseph. "siunitx: A comprehensive (SI) units package". *TUGboat* **32**(1), 95–98, 2011. tug.org/TUGboat/tb32-1/tb100wright-siunitx.pdf.

Wright, Joseph. "siunitx: Past, present and future". *TUGboat* **39**(2), 119–121, 2018. tug.org/TUGboat/tb39-2/tb122wright-siunitx.pdf.

Wright, Joseph. "l3build: The beginner's guide". *TUGboat* **43**(1), 40–43, 2022. tug.org/TUGboat/tb43-1/tb133wright-l3build.html.

◇ Joseph Wright
Northampton, United Kingdom
joseph dot wright (at) morningstar2.co.uk