From \newcommand to \NewDocumentCommand with xparse

Joseph Wright

Abstract

The xparse package provides a new method for creating document macros, moving beyond \newcommand. With xparse it is possible for ordinary LaTeX users to create functions with multiple optional arguments, stars and mixtures of these. This brief article highlights using the xparse approach for the LaTeX user (as distinct from the LaTeX programmer).

1 Introduction

In recent articles, I've been discussing how some of the ideas that the LaTeX3 Project have developed can be used by LaTeX programmers today. However, most users of LaTeX don't want to deal with the programming side: they just want to use LaTeX. The existing LaTeX3 packages can already offer benefits directly to LaTeX users. Here, I want to show how the xparse package (LaTeX3 Project, 2010) can be used to replace \newcommand with a much more powerful way of creating commands for day-to-day LaTeX use.

Before getting started, let me pose the question 'Why would you want to replace \newcommand?' With \newcommand, you can make a macro that takes a number of mandatory arguments, or a macro where the first argument is optional and in square brackets, but that is it as far as variation goes. Anything else then needs the use of TeX programming or internal LaTeX 2$_\varepsilon$ macros: not really helpful for end users. The macros that \newcommand creates are also 'fragile'. This shows up where you need to \protect things, which can be very confusing. Macros created using xparse are robust (*i.e.* not 'fragile'), and are therefore reliable in places like section headings.

2 Getting started with xparse

The xparse package is part of a larger bundle of material (expl3 and xpackages) which the LaTeX3 Project has released to CTAN for general use and distribution. As such, it is included in MiKTeX 2.7, TeX Live 2009, and later releases. If you are using an older TeX distribution you can download both expl3 and xpackages from CTAN, ready to install.

xparse can be loaded as usual for LaTeX 2$_\varepsilon$:

```
\usepackage{xparse}
```

It adds a number of new macros to LaTeX, but here I'll discuss just a few. The main one I'll be using is \NewDocumentCommand, which is the LaTeX3 version of LaTeX 2$_\varepsilon$'s \newcommand.

3 Macros with no arguments

The simplest type of macro is one with no arguments at all. This isn't going to show off xparse very much but it's a starting point. The standard LaTeX 2$_\varepsilon$ method to make a macro with no arguments at all is

```
\newcommand\NoArgs{Text to insert}
```

which with xparse would instead read

```
\NewDocumentCommand\NoArgs{}{Text to insert}
```

That does not look too bad, I hope. Notice that I've got an empty set of braces in the xparse case: this is where the arguments for the new macro would be listed. With \NewDocumentCommand there always has to be a list of arguments, even if it is empty. That's in contrast with the \newcommand approach, where we only need to mention arguments when there are any.

4 Macros with simple mandatory arguments

The most common type of argument for a macro is a mandatory one. With \newcommand, we'd give a number of arguments to use:

```
\newcommand\OneArg[1]{Text #1}
\newcommand\TwoArgs[2]{Text #1 and #2}
```

\NewDocumentCommand is a bit different. Since it can work with different types of arguments, each is individually specified with a letter. A mandatory argument is 'm', so we'd need

```
\NewDocumentCommand\OneArg{m}{Text #1}
\NewDocumentCommand\TwoArgs{mm}{Text #1 and #2}
```

This is still pretty similar to \newcommand: the useful stuff starts when life gets a little more complicated.

5 Macros with one or more optional arguments in square brackets

To get something clever out of xparse, the arguments need to be a little more varied than we've seen so far. Let's look at optional arguments, which LaTeX puts in square brackets. If I want the first argument to be optional, then \newcommand can help:

```
\newcomand\OneOptOfTwo[2][]
  {Text with #2 and perhaps #1}
\newcomand\OneOptOfThree[3][]
  {Text with #2, #3 and perhaps #1}
```

If I want anything else, I'm on my own. First, let's do the above examples using xparse. There, an optional argument in square brackets, as in \newcommand, is specified by 'O' followed by {}:

```
\NewDocumentCommand\OneOptOfTwo{O{}m}
  {Text with #2 and perhaps #1}
\NewDocumentCommand\OneOptOfThree{O{}mm}
  {Text with #2, #3 and perhaps #1}
```

How about two optional arguments? You can't do this with \newcommand. Although it is provided by add-ons like the twoopt package (Oberdiek, 2010), xparse is overall much more flexible. All we need to do is use two O{} statements.

```
\NewDocumentCommand\TwoOptOfThree{O{}O{}m}
  {Text with #3 and perhaps #1 and #2}
```

Then we can do:

```
\TwoOptOfThree{Mandatory}
\TwoOptOfThree[Optional1]{Mandatory}
\TwoOptOfThree[Optional1][Optional2]{Mandatory}
\TwoOptOfThree[][Optional2]{Mandatory}
```

(You can't give only the second optional argument: you still need an empty first one.)

What if we want a default value for the optional argument? With \newcommand, that would be

```
\newcommand\OneOptWithDefault[2][myval]
  {Text using #1 (could be 'myval') and #2}
```

This is where the braces come in: whatever we put inside the braces becomes the default value.

```
\NewDocumentCommand\OneOptWithDefault
  {O{myval}m}
  {Text using #1 (could be 'myval') and #2}
```

The same idea applies to each optional argument: whatever is in braces after the O is the default value.

6 More complicated optional values

You might be wondering why we need the {} after O when there is no default value: why not just o? Well, there is o as well, but it's a bit different. Unlike \newcommand, \NewDocumentCommand can tell the difference between an optional argument that is not given and one that is empty. To do that, it provides a test to see if the argument is empty:

```
\NewDocumentCommand\OneOptOfTwoWithTest{om}
  {\IfNoValueTF{#1}
    {Do stuff with #2 only}
    {Do stuff with #1 and #2}}
```

Don't worry if you forget to do the test: the special marker that is used here will print '-NoValue-' as a reminder!

Sometimes you might want two different optional arguments, and be able to tell which is which. This can be done by using something other than square brackets, often angle brackets (< and >). We can do that using the letter d (or D if we give a default).

```
\NewDocumentCommand\TwoTypesOfOpt{D<>{}O{}m}
  {Text using #1, #2 and #3}
```

What input syntax does this recognize? Let's look at some examples:

```
% One mandatory
\TwoTypesOfOpt{text}
% A normal optional
\TwoTypesOfOpt[text]{text}
% A special optional
\TwoTypesOfOpt<text>{text}
% Both optionals
\TwoTypesOfOpt<text>[text]{text}
```

How did that work? The first two characters after the D are used to find the optional argument, so in this case < and >. The same could be done with (and), or almost anything else you fancy.

Another common idea in LaTeX is to use a star to indicate a special variant of a macro. Creating those with \newcommand is difficult, but it is easy with \NewDocumentCommand:

```
\NewDocumentCommand\StarThenArg{sm}
  {\IfBooleanTF#1
     {Use #2 with a star}
     {Use #2 without a star}}
```

Here, 's' represents a star argument. We see that it ends up as #1, while the mandatory argument is #2. We also need a test to determine if there is a star (\IfBooleanTF). This doesn't mention stars as the test can be used for other things.

7 Summary

There is more to xparse than I've mentioned here, but I hope that this gives a flavour of what it can be useful for. To get more flexibility there is a bit more to think about compared to \newcommand, but the overall consistency is hopefully worth it. By using xparse a whole range of argument arrangements can be supported without needing to know any LaTeX internal functions. This makes the process of creating commands much clearer.

References

LaTeX3 Project. "The xparse package: Generic document command processor". Part of the xpackages bundle, mirror.ctan.org/macros/latex/contrib/xpackages, 2010.

Oberdiek, Heiko. "The twoopt package". Part of the oberdiek bundle, mirror.ctan.org/macros/latex/contrib/oberdiek, 2010.

◇ Joseph Wright
Morning Star
2, Dowthorpe End
Earls Barton
Northampton NN6 0NH
United Kingdom
joseph.wright (at) morningstar2
dot co dot uk