`\starttext`: Practical ConTEXt

Steve Peter

Abstract

In this column, I introduce the reader to the ConTEXt macro package, showing a few practical examples along the way.

1 Introduction

Welcome to the first installment of the `\starttext` column. Together we'll explore the vast world that ConTEXt offers. If you don't already have ConTEXt installed on your system, head over to the Pragma web site at **www.pragma-ade.com**. You can get just ConTEXt, or a complete system with the underlying TEX distribution.

To get the most out of this column, you should have ConTEXt running on your system, and you should type in the examples as we go. You can use any editor that outputs plain text, such as emacs, vi, Text Edit, or Notepad. Don't use a word processor like Word or OpenOffice Writer. There are also complete TEX editing environments like TeXShop on Mac OS X and TEXnicCenter on Windows that allow you to edit your files, run TEX, and view the output from within a single application.

If you don't have TEX, don't have a computer, or are just curious about ConTEXt and would rather read than type, I've also supplied some illustrations.

Let's get started!

2 Hello, World!

Since ConTEXt is a TEX macro package, we'll follow the standard workflow by first entering the text of our document into a plain text file, interspersed with commands that tell TEX to do something with the text (e.g., make it bold, or format it like a footnote). Then we run TEX on the file, and finally we look at the beautiful output.

So fire up your favorite text editor and enter the following:

```
\starttext
Hello, World! This is \ConTeXt.
\stoptext
```

Editor's Note: This article is reprinted, with additions, from *The PracTEX Journal*, 2005(1), `http://tug.org/pracjourn`.

The body of your document is enclosed in a `\start`—`\stop` pair. `\starttext` handles various setup details for you. Save the file as `document.tex`.

If you've used any variety of TEX before, the next step is slightly different, so watch out. (And be amazed!) To run this document through TEX, we'll use `texexec`, a front end script that greatly simplifies life. More about that in a bit. For now, just type the following in a shell window (if you're not using an editing environment as discussed above):

```
texexec document
```

You should now have a new file, `document.dvi`, in your directory. You can view the file with, e.g., xdvi on Unix, TEXniscope on Mac, or yap on Windows. You can convert the dvi (DeVice Independent) file to pdf with the `dvipdfm` utility, or use `texexec --pdf`. Or perhaps your machine may be configured to run pdfTEX automatically (as my machine is). In that case, simply open the resulting `document.pdf` file. Whether dvi or pdf, the result should look something like this:

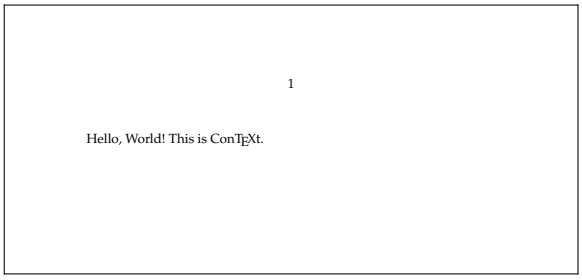

Figure 1 Your first ConTEXt document!

The page number at the top tells us that this is a default ConTEXt document, and not simply a Plain TEX one. For our first experiment, let's put the number into the footer.

Setting up something like the location of the page number is done with a `\setup` command in ConTEXt. Don't worry right now about the exact form of the command. We'll go over them in much greater detail in a later column. For now, to put the number in the footer, add the following line to the top of your document, before the `\starttext`. Run it through `texexec` and look at the file produced.

```
\setuppagenumbering [location=footer]
```

Now the folio is in the footer.

Text of any length is usually subdivided. Let's put in some sections. This time *after* `\starttext`,

put the line

```
\section{First section}
```

Add a few more `\section`s with some text. We'll need them for the next section. To get a bunch of text quickly, try `\dorecurse{20}{\input knuth \par}`.

3 texexec

I mentioned before that `texexec` greatly simplifies life. Why is that? Well, typesetting is a complicated business, and TEX frequently has to collect information on one run to use in a later run. For example, let's add a table of contents. Just after `\starttext`, add:

```
\completecontent
```

But how does TEX know what page the second `\section` is on until after it has typeset the document? The answer, of course, is that it doesn't. TEX gathers up information from all the `\section`s you have in the document and writes that information to an auxiliary file. Normally, you have to then run TEX a second time so that TEX can read that information in and set the table of contents. (And if the TOC is long, it will push everything down, meaning that you have to rerun TEX again!)

Sometimes you find yourself rerunning TEX needlessly just to make certain there aren't any unresolved references. But `texexec` changes that. It automatically reruns TEX as many times as necessary, so you can go refill your coffee.

4 Fun and fancy

Just to whet your appetite, let's take a quick look at a couple of fancier things ConTEXt can do. We'll go into details in future columns. I realize these are a bit of a jump from the basic formatting considered in the other sections, but since we're just setting out, I thought I'd give you a glimpse of some really fancy stuff.

To maintain high typographic standards (cf. the discussion, for example, in Robert Bringhurst, *The Elements of Typographic Style*) you often have to align text, graphics, etc., to a grid, and your text should maintain a consistent position on the baseline grid. Add this to the top of your document and process it with `texexec`.

```
\setuplayout[width=middle,location=middle,
     grid=yes,marking=color]
\moveongrid[both]
\showgrid
```

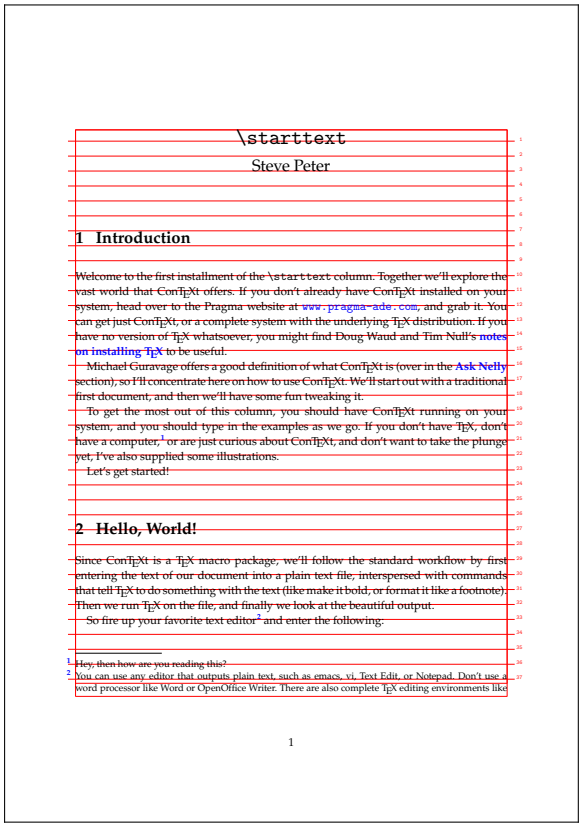
\starttext
Steve Peter
1 Introduction
Welcome to the first installment of the \starttext column. Together we'll explore the vast world that ConTEXt offers. If you don't already have ConTEXt installed on your system, head over to the Pragma website at www.pragma-ade.com, and grab it. You can get just ConTEXt, or a complete system with the underlying TEX distribution. If you have no version of TEX whatsoever, you might find Doug Waud and Tim Null's notes on installing TEX to be useful.
Michael Guravage offers a good definition of what ConTEXt is (over in the Ask Nelly section), so I'll concentrate here on how to use ConTEXt. We'll start out with a traditional first document, and then we'll have some fun tweaking it.
To get the most out of this column, you should have ConTEXt running on your system, and you should type in the examples as we go. If you don't have TEX, don't have a computer,[1] or are just curious about ConTEXt, and don't want to take the plunge yet, I've also supplied some illustrations.
Let's get started!
2 Hello, World!
Since ConTEXt is a TEX macro package, we'll follow the standard workflow by first entering the text of our document into a plain text file, interspersed with commands that tell TEX to do something with the text (like make it bold, or format it like a footnote). Then we run TEX on the file, and finally we look at the beautiful output.
So fire up your favorite text editor[2] and enter the following.
[1] Hey, then how are you reading this?
[2] You can use any editor that outputs plain text, such as emacs, vi, Text Edit, or Notepad. Don't use a word processor like Word or OpenOffice Writer. There are also complete TEX editing environments like
1

Figure 2 The matrix? No, it's the grid.

If you've ever tried to do that with Plain TEX, or even LATEX, you know what a pain it can be. However, ConTEXt does it easily, and even shows you where the grid is, so you can debug troublesome documents.

After all TEX is, when you get down to it, a programming language. That means at some point you'll need to debug your documents. The grid feature is but one of several nice visual debugging tools provided with ConTEXt. For another one, add this to the beginning of your document to gain a view of how TEX puts boxes and glue together:

```
\showmakeup
```

In figure 3 we can see the bounding box for the E in TEX, along with the negative kerns, shown as the thicker boxes near the base of the E.

One more useful visualization command shows you the layout on the page of your text block, mar-

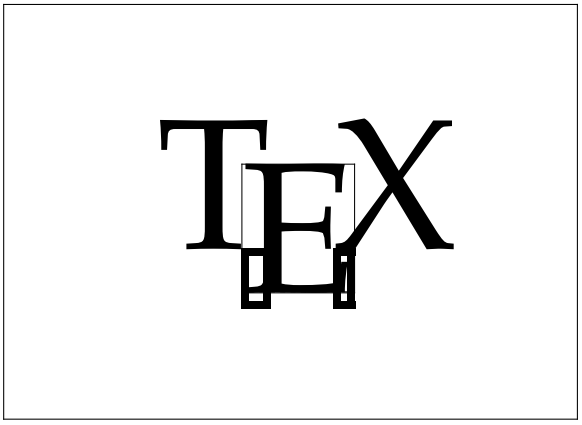

Figure 3 What TEX might look like to TEX

gins, headers and footers:

```
\showframe
```

For more on this aspect of visual debugging, see Hans Hagen's paper in *TUGboat* vol. 19, no. 3.

You don't need a fancy commercial page layout program to set crop marks or do imposition. Put these lines before `\starttext`, run `texexec`, and watch the magic! (Figure 4.)

```
\setuppapersize [A7][letterpaper]
\setuparranging [2*2,rotated,doublesided]
\setuppagenumbering [
     alternative=doublesided]
\setuplayout [margin=0pt,width=fit]
\setupbackgrounds [text][text][
     background=screen]
\setupcolors [state=start]
\setuplayout [location=middle,
     marking=color]
\setuptolerance[tolerant]
\setupbodyfont [palatino,6pt]
```

You can even produce a negative by replacing the first line above with

```
\setuppapersize [A7][letterpaper,
     negative,mirrored]
```

I won't show it here, due to obvious ink costs. But if you ever need to generate film output, this is a lifesaver.

5 Links

I hope you've enjoyed this first look at ConTEXt.

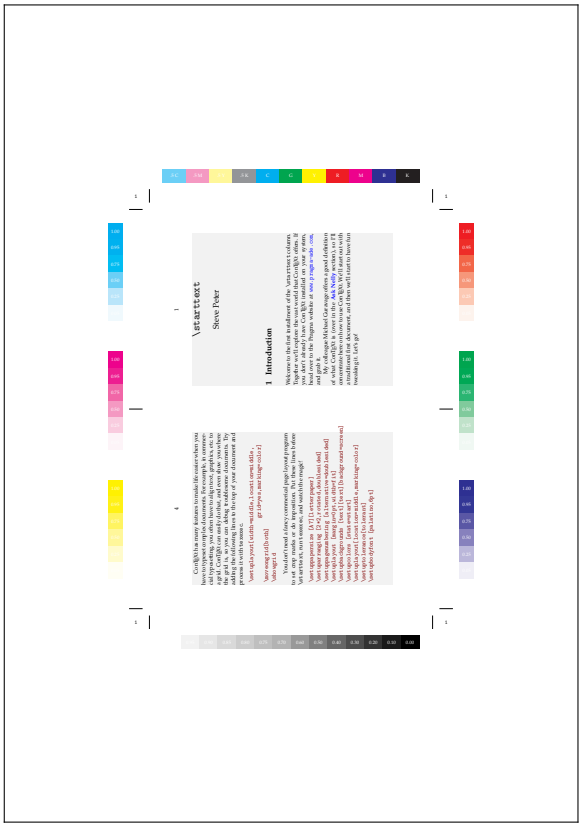

Figure 4 Imposition with ConTEXt

There are numerous topics we haven't addressed yet, such as cross references, hyperlinks, indexes, MetaPost figures and other graphics, and ConTEXt's incredible support for pdf trickery.

There's a lot of information out there and plenty to explore. Start with the documentation on the Pragma web site (`www.pragma-ade.com`).

For examples, check the ConTEXt wiki at `contextgarden.net` and work your way through Bill McClain's excellent page detailing ConTEXt at `home.salamander.com/~wmcclain/context-help.html`. Last, but certainly not least, you can jump into the never-ending discussion on the official mailing list at `www.ntg.nl/mailman/listinfo/ntg-context`.

Join us here in future issues of *TUGboat* for more on the practical use of ConTEXt.

◇ Steve Peter
Beech Stave Press,
310 Hana Road,
Edison, NJ 08817
`speter@beechstave.com`