Speaking TEXnically

How the OECD Test Results were Produced

Amy Hendrickson, TEXnology Inc.

LATEX is a batch processing system. It is a Turing Complete programming language, and was originally developed to produce handsome math books, written by Donald Knuth at Stanford.

Either XeLaTeX or LuaLaTeX, newer implementations of LATEX, are needed to be able to use the excellent font software, fontspec.sty, which enables us to use any font for which we have an .otf file, thus any font in the Adobe font library may now be used in a TeX document.

The particular LATEX system I used, LuaLaTeX, processes the .tex file straight to .pdf. In my testing, LuaLaTeX produced better images when .pdf files were embedded in the illustrations than XeLaTeX did.

The LATEX language, along with TikZ, a sort of meta-PostScript, gives us the ability to describe the format of the document, use the actual text in a macro, and use tools like loops, conditionals and counters to do programming that is needed to produce the unique graphics.

This allows us to reproduce the design of the OECD e-books, and completely automate the production process.

Step 1: Implement the design

The first step involved imitating the elegant design, as seen at the OECD site.

Although the LaTeX language was used for producing the booklets, as you can see the rich visual language used in the original necessitated borrowing many images from the sample .pdf, and then positioning the images in the process of building up the pages.

Step 2: Parse the Data

The basic mechanism that allows us to parse the data and produce LaTeX definitions for each bit of data, is the LaTeX construction `\csname ...\endcsname`. This useful tool allows us to make definitions that contain numbers within their name.

For instance: `\def\csname 123\endcsname{--text--}` is a valid definition, which can be called using `\csname 123\endcsname`. In this case the results would be '`--text--`'. The Very useful consequence is that we can use `\csname...\endcsname` to dynamically make definitions, by advancing a counter each time a new definition is made and using the current state of the counter in the name of the new definition.

This tool is used to parse the data in the .csv file and produce definitions, with unique numbers built into the definition name. This allows us to store data in the body of the definition, which can be later called using the unique set of numbers for the current school and number for the current graphic.

Each school is assigned a number; each data point is also assigned a number. Using the combination of these two numbers we can access the data needed for the current graphic.

The Details: How this tool is used

In the data, each school is assigned a number:

```
%% School numbers:
%   22 Horizon
%   23 Cactus High School
%   24 Ironwood High School
%   25 Liberty High School
%   26 Centenial High School
...
%   36 Metropolitan Learning Center
```

This allows us to build a set of unique macros that use the school number and the entry number in the name of the definition. For example:
`\expandafter\gdef\csname school22-2\endcsname{USA}`
will give us a definition named `\csname school22-2\endcsname` which in this case is defined as 'USA'.

Once we have this definition made, we can access that information any time that
`\csname school22-2\endcsname`
is used. By changing the number of the school, we will access a new definition that contains information appropriate for the current school.

For instance `\csname school23-2\endcsname` will give us information about Cactus High School, and `\csname school26-2\endcsname` will give us information about Centenial High School (see list of school numbers above).

The parsing macros, written in LaTeX, will process the .csv file(s) and send definitions out to an external file, in this case called data.dat. The definitions look like this:

```
\expandafter\gdef\csname school22-2\endcsname{USA}
\expandafter\gdef\csname school22-3\endcsname{AR}
\expandafter\gdef\csname school22-4\endcsname{ARIZONA}
\expandafter\gdef\csname school22-5\endcsname{}
\expandafter\gdef\csname school22-6\endcsname
                              {PARADISE VALLEY SCHOOL}
\expandafter\gdef\csname school22-7\endcsname{22}
\expandafter\gdef\csname school22-8\endcsname
                                 {HORIZON HIGH SCHOOL}
\expandafter\gdef\csname school22-9\endcsname{City
  (100,000-1,000,000 people)}
\expandafter\gdef\csname school22-10\endcsname{Public}
\expandafter\gdef\csname school22-11\endcsname{2152}
\expandafter\gdef\csname school22-12\endcsname{13}
\expandafter\gdef\csname school22-13\endcsname{5}
\expandafter\gdef\csname school22-14\endcsname{14}
...
```

When we reach the number 274, we start with a new school, and start using the new school number, in this case 23, and start with entry # 2:

```
\expandafter\gdef\csname school22-273\endcsname{0.41}
\expandafter\gdef\csname school22-274\endcsname{0.14}
%% new school starts here
\expandafter\gdef\csname school23-2\endcsname{USA}
\expandafter\gdef\csname school23-3\endcsname{AR}
\expandafter\gdef\csname school23-4\endcsname{ARIZONA}
\expandafter\gdef\csname school23-5\endcsname{}
\expandafter\gdef\csname school23-6\endcsname{PEORIA UNIFIED
 SCHOOL DISTRICT}
\expandafter\gdef\csname school23-7\endcsname{23}...
```

And this process continues until all the entries in the .csv file are translated into definitions.

A number of .csv files are used. One is for data for bubble graphs, so a new .sty was built to parse the bubble .csv file. You can see the results of the parsed bubble data below where a 'bubnum' counter as well as the school number is used, as well as the entry number.

```
\expandafter\gdef\csname bubnum1-school22-2\endcsname{22}
\expandafter\gdef\csname bubnum1-school22-3\endcsname{USA}
\expandafter\gdef\csname bubnum1-school22-4\endcsname{82}
```

Prepared data files

When all the data for this project has been parsed, we end up with four data files, that each contain LaTeX definitions.

Step 3: Prepare root .tex file

To simplify production, there is a macro defined for each individual page in the e-book.

Embedded within the definition for pages where unique graphics appear are commands that use the `\csname...\endcsname` form, allowing us to call for the data for the current school.

Below is an example of the root file. As you can see near the top of the file a number of other files are brought in, the most important being amy.sty which has the definitions for all the individual pages, with calls for the data for each school embedded. amy.sty has 16,687 lines of code.

The .dat files are the data definitions that have been previously prepared when the .csv files were parsed.

```
\documentclass{bookx}
\usepackage{amy}
\usepackage{amsbsy}

\input bubdata.dat
\input ./specs4/data.dat
\input cdata.dat % country data
\input ./csv2/varidata.dat % for page 16

\begin{document}
%% use any number from 22 to 36:
\schoolnum=25

{\cover}
{\titlepage}
{\copyrightpage}
{\pagethree}
{\pagefour}
{\pagefive}
...
{\pageonesixtytwo}
```

How are the new commands used?

We have a set of unique definitions that have counters built into their name, and now we can call those definitions.

In order to know which number to use when making the macro for a particular page, CTB/McGraw-Hill provided the very useful TestSchoolLayout file that showed what data entry number was needed in a particular graphic.

Here is an example of the calls for data for a particular school, in this case for figure 2.1b, the top example on page 3, above.

```
\def\cs#1{\csname school\the\schoolnum-#1\endcsname}
%% <<=== definition to make calling for the
%%       csname definition easier
...
\hbox to\textwidth{\hskip-10pt
\hskip3.5pt\includegraphics{samp3plainb.pdf}\hfill}
\vskip18.8pt
\vbox to0pt{\vss
\hskip169pt$\vcenter to 0pt{\vss
\hsize=7pt
\xgraysquare\vss}$
\vskip\cs{21}\yunit
%% <<<=== Here is where the school data is used.
                        %%(\cs is defined above)
\vskip-.5pt
}
...
```

When a background illustration is used, we need to determine the 'x' and 'y' values, by measuring the graphic carefully and using iterative testing against the background to find the size of the horizontal and vertical units. This will give us the value for `\xunit` and `\yunit`, which you see used in the example above. This code:

```
\vskip\cs{21}\yunit
%% <<<=== Here is where the school data is used
```

is where the vertical positioning of the bar is determined. The positioning will be different for each school, since the definition for data entry #21 will be different for each school.

Evaluation of the process

Parsing the data is easy, but to position the data in the graphs, and to do the back ground programming necessary to determine whether a particular marker is statistically significant and should be colored differently, can be painstaking.

However, the time invested is compensated in that it only takes only a bit more 3 minutes to run off a unique ebook, a process that could be repeated 1000 times if there was data for 1000 schools. In a quick test, 13 unique booklets, each 162 pages long with hundreds of graphs and graphic images, were processed in 45 minutes, averaging 3.5 minutes per booklet.

Questions, Comments? Send email to amyh@texnology.com

***

Amy Hendrickson
amyh@texnology.com
www.texnology.com