Creating ASCII Graphics with cadubi, figlet, and boxes

Letter Art

Friendly frames around email messages, colorful login prompts and three dimensional fonts – and all of that just using simple letters. ASCII art has been around for years, but producing such images has always been hard work. The tools we will be looking into in this article reduce the effort, leaving you plenty of time to work on the creative side of your designs.

BY ANDREAS KNEIB

A mere couple of thousand years after inventing the alphabet, mankind started to alienate it. In the days of green on green monitors and character-based consoles, what is commonly known as **ASCII** art was invented; this is an art form that involves using letters to paint pictures. While the text editors of bygone years meant that you had to individually position each letter, intuitive tools now help modern-day ASCII artists.

ASCII graphics typically do not need to use special characters. This makes them suitable for informative graphics in cross-platform applications and on older machines (see Figure 1). But above all, painting by letters is a fun thing to do. Taking the trouble to dress up ***/etc/issue*** or ***/etc/motd***, for example, adds that personalized touch to the login process. We will looking at the details in this article.

The Calligrapher

Let's start off with *figlet* [1], a tool that conjures up all kinds of font images, like the Debian logo in Figure 2. Debian users can simply type *apt-get install figlet* to install the calligrapher on their hard disks. *apt-get* will also install to the *cadubi* and *boxes* utilities, which we will be looking at later. Refer to Box 1 for more details about installing the tools on your distribution.

But let's get back to *figlet*. The program displays your text input as ASCII banners using fonts that are made up of single characters. This makes them ideal for command line use, as the command line only has a single fixed width character set.

figlist displays an alphabetically sorted list of the fonts available with *figlet*. You might like to send the command to the *less* pager, unless you are into speed reading, of course:

```
> figlist | less
Default font: standard
Font directory:/usr/share/figlet
Figlet fonts in this directory:
3-d
3x5
[...]
big
[...]
```

The *showfigfonts* command displays examples of the *figlet* fonts.

```
01 > showfigfonts | less
02 [...]
03 big :
04  _     _
```

GLOSSARY

ASCII: *The "American Standard Code for Information Interchange" provides a character set with the letters of the alphabet for computer programs. Type man ascii for more details.*
/etc/motd: *The "Message of the Day" is displayed to users when they log on.*
/etc/issue: *This file is displayed on the console before the login prompt appears and typically contains notes on the distribution and kernel version.*

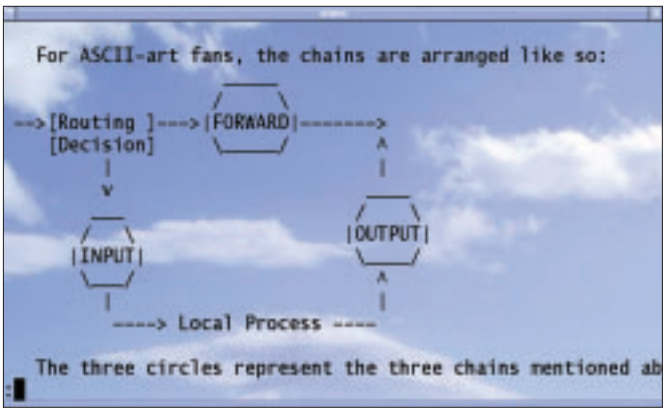

Figure 1: An informative graphic in ASCII format

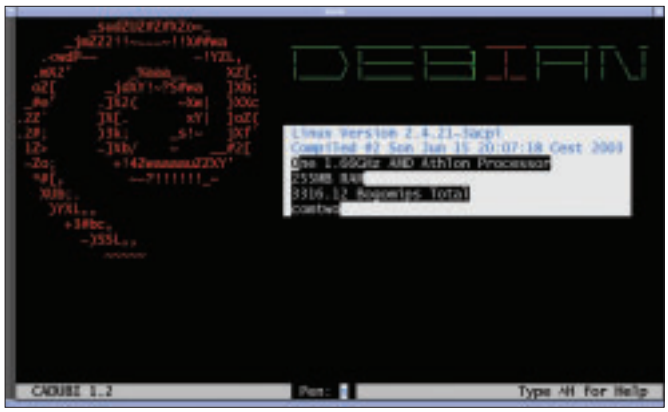

Figure 2: cadubi creating an /etc/motd

```
05 | |   (_)
06 | |__  _  __ _
07 | '_ \| |/ _` |
08 | |_) | | (_| |
09 |_.__/|_|\__, |
10           __/ |
11          |___/
12 [...]
```

The fonts that *showfigfonts* and *figlist* discovered are passed to *figlet* using the *-f* flag, and followed by the text you want to display. Thus, *figlet -f caligraphy Linux* will write the word *Linux* in old-fashioned, manuscript type letters.

figlet normally line-wraps after 80 characters, but *-t* will automatically adjust this to your terminal's line width. The *-w* parameter allows you to specify the width of the text; for example, *-w 40* will line-wrap after 40 characters.

The *-c* (centered), *-l* (left) and *-r* (right) parameters specify the text alignment. *-R* will even print your text from right to left.

As is the case for nearly any command line tool, you can use pipes here. In our next example, *tr* replaces the hash signs (*#*) in the *banner3* font with at characters (*@*) (see Figure 3):

```
figlet -f banner ➥
ABC | tr '#' '@'
```

The manpage for the package details typographic niceties, such as kerning and the like. Type *man figlet* to access the manpage.

The Frame Maker

Now let's add a fitting frame to the decorative font: this is not part of the mini DTP program *figlet*'s portfolio, so it simply hands the job over to a specialist. *boxes* is the tool that will place a decorative frame round our text art, again in pure ASCII (see Figure 4).

The *-l* (list) option tells you the frame styles the program has to offer. Again it makes sense to output the list page-by-page:

```
boxes -l | less
```

The *-d* parameter selects a style:

```
boxes -d boxname old new
```

old is the name of the original file and *new* the name of the framed output file. But the tool will also use the standard input and output of your terminal:

```
andreas:~> echo "This is
> a C comment"|boxes -d c-cmt2
/*
*  This is
*  a C comment
*/
```

This is one of *boxes*' more practical applications. The tool not only creates

Figure 3: Figlet fonts can be manipulated on-the-fly using the "tr" comand

Box 1: Installation Notes

cadubi is available at [3]. First unpack the tarball by typing *tar xvzf cadubi_1.2.tar.gz*, and *cd* to the *cadubi* directory. Then launch the Perl script by typing *./cadubi*. *cadubi* requires Perl 5.002 or later, and also the Perl *Term::ReadKey* module. The *README* file in the *TermReadKey-2.21.tar.gz* package tells you more about how to install the module, as does the *perldoc -q cpan* command.

The source code, RPM and Deb packages for *figlet* are all available at [1]. Use your distribution's tools to install the packages. If you prefer to compile *figlet* yourself, first type *tar xvzf figlet221.tar.gz* to unpack the tarball. By default, *figlet* will install programs, scripts, fonts and the manpage in */usr/local*; to change the default, edit the *DESTDIR*, *DEFAULTFONTDIR*, and *MANDIR* variables in the Makefile. You will also need to comment out the

```
DEFAULTFONTDIR = fonts
```

line using a hash sign *#*, as the font installation will not work otherwise. If the */usr/local/man/man6* does not exist (this is where Figlet attempts to copy the manpage), type *mkdir -p /usr/local/man/man6* to create it.

make figlet creates the binaries, *figlet* and *chkfont*. Now all you need to do, is to make the *figlist* and *showfigfonts* shell scripts executable; type *chmod +x figlist showfigfonts* to do so. *su* to *root* before calling *make install* to install the program components in the appropriate directories.

To install *boxes*, use the pre-compiled RPM package, which is available on the homepage [2].

Table 1: Cadubi's Color Palette

Value	Color
0 or N	Default
1 or W	White
2 or R	Red
3 or G	Green
4 or Y	Yellow
5 or B	Blue
6 or M	Magenta
7 or C	Cyan
8 or K	Black

Figure 4: An impressive pair: the figlet font program and the frame generator, boxes

decorative frames, but also comment styles for various programming languages. To leverage this function, it makes sense to integrate *boxes* with your favorite text editor. The boxes project site tells you how to do that with Jed and the usual suspects, Vim and Emacs, besides providing more tips on designing your own frames [2].

In the case of larger boxes (e.g. the *dog* frame style in Figure 4) with very little content, it makes sense to define the alignment. The *-a* option introduces the alignment, and is followed by *c*, *r* or *l* (for centered, right, and left).

The padding option (*-p*) defines the distance between the text and the frame; you can add a direction and a number of characters. For example, *-pa5t3r1*, which puts the content five characters away from the frame in any direction (*a5*). The distance to the top margin is only three lines, (*t3*) however, and the distance to the right margin only one character (*r1*). The following values can be used with *-p*:

- *a*: In *a*ll directions
- *b*: From the *b*ottom of the box
- *h*: In *h*orizontal direction
- *l*: From the *l*eft side of the box
- *r*: From the *r*ight side of the box
- *t*: From the (*t*op) of the box
- *v*: In *v*ertical direction

A fixed frame that is independent of the text content can be created using the *-s* (*s*ize) option. The following example creates a box that is nine characters wide and three lines tall:

```
> echo LINUX | boxes -s 9x3 ➥
-d shell
#########
# LINUX #
#########
```

The final important parameter is *-r* (for *r*emove). This allows you to remove any boxes you defined incorrectly, or do not like, without affecting their content. *boxes* automatically recognizes the frame style, and successfully removed even double frames in our test.

The Painter

Our experiments so far have been staid, monochrome affairs. But the ASCII Art Editor *cadubi* [3] brings a little color to the scene.

Anyone who has done battle with Escape sequences [4] trying to define a colored prompt, might be surprised to discover how easily *cadubi* copes with this task. In contrast to its distant siblings, the editors, *cadubi* actually provides WYSIWYG output. You choose the colors, and *cadubi* adds the Escape sequences in the background to tell the Shell to display colored text.

Figure 5: Logging in is a lot more fun, if you have your own colorful */etc/motd*

The tool has two operating modes: a text input mode, which you enter by pressing [t], and a command mode that is entered by pressing [Esc].

The bar at the bottom of the screen shows you the current mode and the selected color in the *Pen:* section (see Figure 2).

In text input mode, the program is much like any other text editor: pressing a key, creates a character. But in command mode, you can use the keyboard to define foreground and background colors, open and save files, or read the online help.

After pressing [Esc] to change to command mode, you can use the arrow keys to move the cursor. If your arrow keys do not work, you can use the [i], [j], [k] or [l] keys for up, right, down, and left.

Control keys are assigned to important commands. The keyboard shortcut [Ctrl-h] opens up a help page, and [Ctrl-w] re-draws the screen. You press [Ctrl-r] to open a file, and [Ctrl-o] to save. [Ctrl-x] quits the program.

Colors are selected by pressing the [f] (foreground) and [b] (background) keys. *cadubi* then prompts you for the color in the status line. Table 1 provides an overview of the shortcuts that represent the available colors. Besides colors, there are additional elements, such as bold type [g], inverse output [v] or blinking text [W]. [p] changes the *Pen Character*; this is similar to a brush in command mode. The space key adds this to the selected color combination – this is ideal for painting larger areas.

It pays to use the *cat* command to check your work before publishing it as a hand-painted message for other users in ***/etc/motd*** (Figure 5). ■

INFO

[1] Figlet: *http://www.figlet.org/*
[2] Boxes: *http://boxes.thomasjensen.com/*
[3] Cadubi: *http://www.logicallemon.com/projects/cadubi/*
[4] Escape sequences: *http://www.tldp.org/HOWTO/Bash-Prompt-HOWTO/x343.html*