Hacking on the *mg* macro package

John Ankarstr m

What is *mg*?

Mg is a simple macro package for troff designed to abstract as little as possible from troff itself, while still providing a powerful framework for writing advanced documents.

How is the source code of *mg* organized?

If you run `grep -n [-]-` on the *g.tmac* source file, you are presented with an overview of *mg*'s macros:

```
29:.\" Internal macros ---------------------------
31:.\" @a -- setup document
119:.\" @c -- copy environment
127:.\" (e -- set environment
132:.\" @e -- set extended environment
174:.\" @f -- footer
180:.\" @h -- header
187:.\" @tf -- footer trap
202:.\" @th -- header trap
211:.\" @tn -- footnote trap
233:.\" Inline macros -----------------------------
235:.\" b -- bold font
240:.\" c -- constant-width font
266:.\" i -- italic font
271:.\" x -- bold italic font
277:.\" Hybrid macros -----------------------------
279:.\" q -- quotation
294:.\" Environment macros ------------------------
297:.\" d -- centered date
304:.\" h -- heading
311:.\" l -- literal display
318:.\" p -- paragraph
325:.\" s -- subheading
332:.\" t -- centered title
339:.\" Other macros ------------------------------
341:.\" ( -- begin footnote
356:.\" ) -- end footnote
376:.\" w -- want space
```

This is a sufficient summary of the entire *mg* source code, as nothing is performed outside of these macros. All initialization is performed in the `@a` macro, which is automatically called at the first invocation of any other macro.

The above summary reflects a categorization in the macros defined by *mg*. There are internal and external macros. The former are to be used within *g.tmac* itself, while the latter are to be used in *mg* documents. Among the external macros, there are inline, environment (or block-level), hybrid and other macros.

The inline macros all follow the same pattern. They take three arguments: the string to be formatted, an optional suffix and an optional prefix. The hybrid macros act as inline macros when given arguments; otherwise they act as environment macros.

The environment or block-level macros generally take no arguments (except `d`). Instead, they activate a given environment, affecting the formatting of the following text. Each environment macro is associated with a specific environment, carrying the same one-letter name as the macro itself.

As you can see, the macros in each category are arranged alphabetically.

Where is document state stored?

Most state is stored by troff itself within the different environments. In addition, *mg* associates three extra registers with each environment: `sp`, the amount of space to be added by `@e` before an environment; `sq`, the same (except the space is not added if the new environment is identical to the previous one); and `ti`, the indentation of the first line in the `p` environment. These are stored in registers named `@ENV_sp`, `@ENV_sq` and `@ENV_ti`, where `ENV` is the name of the associated environment.

The strings `%env` and `%penv` contain the name of the current and previous environment.

The `@a` register is set to 1 if the document has been initialized (i.e. if `@a` has been invoked).

The `@m` register is non-zero if “manual footer” mode is active. If `@m` is non-zero, `@tf` decrements it by one and exits when invoked, unless called with the `f` (force) argument. This is useful if you want to trigger the footer manually, but do not want the printed footer to trigger the footer trap again.

`@.t` contains the absolute vertical position of the first trap following the first footnote reference on a page; it is set and used by `)` to place the footnote trap in the correct vertical position. `@dn` contains the height of all collected footnotes on a page; it is set by `)` and reset to zero by `@tn`. `@n` contains the total number of collected footnotes.

Note that none of these registers and strings should be directly accessed or modified by *mg* documents.