The DOCGEN documentation generator

Stephen Pelc
MicroProcessor Engineering
133 Hill Lane
Southampton SO15 5AF
England

Tel: +44 (0)23 8063 1441
Fax: +44 (0)23 8033 9691
Net: sfp@mpeltd.demon.co.uk
Web: http://www.mpeltd.demon.co.uk

DOCGEN is a system for generating manuals in HTML or Tex from Forth source code. It has been used at MPE to produce all the manuals for ProForth VFX for Windows, and for many applications. This paper discusses how DOCGEN works, the results produced, and its future development

Problem

Producing documentation is very few programmers' favourite activity. Keeping the documentation accurate and up to date is even less absorbing. When MPE produced ProForth version 2.0, writing the manual cost more than writing the code. With ProForth VFX for Windows we were determined to reduce the cost of documentation, and to ensure that an accurate up-to-date manual is available for each release of the product.

We also noted that, with the increasing convergence between the PC product kernel and the embedded target systems, we are increasingly using the same code on several different systems, each of which has to be documented.

We note too that the only real reference to a piece of software is the source code.

Solution

The solution to this problem is to put the documentation in the source code. With the availability of multiline comments, MPE's programmers were increasingly putting design information in the code. What we needed to do was to be able to expand this so that it was suitable for use as printed documentation.

We also needed to be able to produce documentation for our embedded target code and for applications produced for legacy systems.

There are several literate programming tools available, both commercially and as shareware, but these suffer from being separate tools. We want to ensure not only that documentation can be produced at compile time, but also that DOCGEN is an integral part of the supplied system for users of ProForth VFX. Previous experience with other literate programming packages

during a large project written in C showed that the use of conventional tools, especially those that permitted external editing of the documentation section, leads to inaccuracy and eventual abandonment of literate programming. We therefore wrote our own tool, despite our misgivings about reinventing the wheel. What is presented here is the result.

DOCGEN is a system which parses formal comments in Forth source code and produces documentation from it.

In order to be compatible with existing code, DOCGEN is based around the lowest common denominator Forth comment:
(…)

In order to be considered as a DOCGEN line the comment must start at the left hand column, and after one space must contain ***X** where X is a command indicator. The following is taken from the DOCGEN source file itself.

```
( *> docgen                                                          )
( **                                                                 )
( *S In-Source Formal Comments                                       )
( **                                                                 )
( *P DOCGEN output is derived from formal comments within the source    )
( ** code. Output format is standard HTML 2.0.                       )
( ** The comment takes the form:                                     )
( **                                                                 )
( *E - *x blah blah                                                 )
( *P Where the open bracket must be in column zero, X is the operation  )
( ** code and "blah blah" the control text.                          )
( **                                                                 )
( *H Valid Operation codes are:-                                     )
( *D *  The following text is a continuation for the current style. This)
( **    can only be used after a G,E or P operator.                  )
( *D !  Create and select a new output file. The control text is the    )
( **    filename without extension.                                  )
( *D >  Select and append to existing output file. The control text is  )
( **    the filename without extension.                              )
( *D T  Following text is a section title.                           )
( *D S  Following text is a section sub-title.                       )
( *D N  Following text is a section sub-sub-title.                   )
( *D D  Following text is a definition. The first space delimited token )
( **    is the term, the remaining text the description.             )
( *D P  Begins a new paragraph.                                      )
( *D E  Begins a paragraph which is a code example.                  )
( *D B  Following text is a bulleted entry.                          )
( *D G  Following text is a glossary entry. The preceeding line is      )
( **    output in a fixed font code format.                          )
( *D R  Following text is output directly when using the TeX output.    )
( *D W  Following text is output directly when using the HTML output.   )
( *D H  A simple heading.                                            )
( *D C  A fixed font line.                                           )
```

DOCGEN can produce HTML directly for on line documentation, or can produce TeX output which can then be processed to produce PDF files. Which format is used depends on the requirement for the documentation. If on-line documentation is to be produced, HTML is the natural choice. If printed documentation is required, then a PDF file is the natural choice.

The following shows the documentation for the word PARSED which is used to produce documentation for a source file but does not compile it. This word is heavily used for processing embedded system code which could not be compiled under ProForth VFX.

```
: PARSED     \ c-addr u --
( *G Similar to INCLUDED but performs no actual compilation. This       )
( ** allows formal comments to be parsed from a source which you do not )
( ** want compiled.                                                     )
```

The result of this (as far as Microsoft Word will allow) is:

: PARSED \ c-addr u –
Similar to INCLUDED but performs no actual compilation. This allows formal comments to be parsed from a source which you do not want compiled.

Results so far

The whole of the ProForth VFX for Windows manual has been produced using DOCGEN. DOCGEN is an easy way to produce up to date and accurate documentation. However, it isn't perfect yet! That having been said, it is MPE's preferred method of producing documentation, albeit at the cost of larger source files. Given the size, speed and price of modern hard disks this is a small price to pay.

Of particular benefit is the ability to produce glossary entries that are easy to update. When a user calls technical support for help, we can change the documentation immediately, say to add an example, and be sure that the update is reflected in the next release.

The HTML output requires one hand edit of INDEX.HTM because the assumption is made that all pages are referenced from another page.

You still sometimes need to know which output format is being generated.

In order to insert the '(' or ')' characters inside your text, you need to use ASCII characters 30 and 31 respectively, and not all program editors accept these. These need to be made user definable or an escape character such as '\' or '`' needs to be used.

How DOCGEN works

DOCGEN works by installing two hooks inside the ANS word **REFILL**. The words are **DOCGEN_PREREFILL** and **DOCGEN_REFILL** which are executed before and after the new text is read. These are **DEFER**red words with a default action of **NOOP**. Switching on DOCGEN assigns new actions to these words.

In **DOCGEN_PREREFILL** the last line is saved in case the next line is a glossary tag *G and the current line is processed to handle tabs and other special characters. **DOCGEN_REFILL** checks to see if this is a formal comment (*X at column 3) and processes the comment if it is a formal comment.

The first version of DOCGEN was written as an ANS compliant application which could be added to any ANS Forth. The current version has departed from this goal because of the facilities available in ProForth VFX. However, restoration of pure ANS code source is not difficult.

Future developments

The next version of DOCGEN will include the following new features.

User definable profiles
The user can add generators for different word processors or file formats.

Extensible tags
The user will be able to override the action of a tag or add a new one.

Quick reference guide generation
Glossary entries in MPE code include a tag to say which section of a specification it comes from. For example, all the ANS words are tagged with their ANS reference number. By using these tags as identifiers, we can sort the output of DOCGEN to produce quick reference sheets that contain word names, stack comments, and a brief description.

Special character handling
As mentioned earlier, DOCGEN needs to be able to handle the '(' and ')' special characters in the text, and to be able to use additional special characters would be useful.

Conclusions

Producing Forth documentation and manuals directly from source code is not only possible, but also provides a comfortable working environment. DOCGEN has enabled MPE to produce up to date and accurate manuals for ProForth VFX much faster and at much lower cost than using a separate manual.

It is unlikely that we shall write new software manuals using conventional word processors again.

Acknowledgements

The original design of DOCGEN was done by Steven Coul. Further work on DOCGEN has been performed by Edward Bell and Stephen Pelc.

Papers by and conversations with the following people have been useful:
Anton Ertl, Peter Knaggs