Sample Literate Programming

BY SAM LIDDICOTT

Using TEXMACS and Fangle

Abstract

Literate Programming will not be explained here but it will be shown as a way narrate program development showing the progression of ideas a is natural for humans to understand, leaving the generation of programs from those ideas to the machine.

TEXMACS is used as the editor and FANGLE[1] as the extraction tool.

Here we will explain the popular *hello world* written in C.

1 Message

Here is the message that we wish to give to world:

1a ⟨message[1](), lang=txt⟩ ≡

```
Hello World!
```

Used by 1b, 1c

Because we did not press enter in the literate programming environment, this line is not prefixed in the document tree with an `<item>` tag and so has no line number displayed and is not a line but a line-fragment.

Note 1. The header contains the chunk name *message* as well as the language of the listing which is *txt*. The beginning of the header gives the id to this *chunk* of code — 1a — which suggests to the reader that it is the first chunk on page 1.[2]

2 Displaying the message

We have a few choices available. The message can be output with `printf`, like this:

1b ⟨message-printf[1](), lang=cpp⟩ ≡

```
1 printf("⟨message 1a⟩\n");
```

Used by 1d

or even with `puts`:

1c ⟨message-puts[1](), lang=cpp⟩ ≡

```
1 puts("⟨message 1a⟩\n");
```

not used

But, we prefer `printf` which is more traditional.

Note 2. These last two chunks have a different letter for the chunk reference even though they may be on the same page.

Note 3. They also contain a line number, because they are intended to be a full line of text.

3 The main function

Note 4. If a code chunk is split across two pages, as this next code chunk is, the code chunk header is repeated in the page header of the last page. The technique used is to put a TEXMACS *label* at the begining of the chunk and also at the end of the chunk. We take the *pagerefpage* of each label, and if they are not on the same page, then at the end of the chunk we set the page header to be the same as the chunk header.

We have to enclose this line in the standard C *main* function, like this:

1d ⟨main[1](), lang=cpp⟩ ≡

```
1 int main(int argc, char** argv) {
2   ⟨message-printf 1b⟩
```

1. http://new.fangled.org

2. The label and hyperlink 1a were generated with the TEXMACS document tag `<chunk-reference|message|1>` to generate a link to the first chunklet of the chunk named *message*.

1d ⟨main[1](), lang=cpp⟩ ≡

```
3   return 0;
4 }
```

Used by 2a

The statement `return 0;` lets the operating system know that the function completed successfully — which is a bit of a presumption as we don't check if `printf` is successful.

Note 5. Note the `printf` statement defined in 1b has been included in 2.

Note 6. The line numbering starts counting again from 1 for each named chunk.

4 Composing the file

By convention, with TEXMACS and FANGLE, a chunk whose name begins with a `./` is considered to be a file suitable for automatic extraction; so we will define out main chunk to be `./hello-world.c`

Header files

On my system, both `printf` and `puts` both require the header `stdio.h`; so this line becomes the first line of our file.

2a ⟨./hello-world.c[1](), lang=cpp⟩ ≡ 2b▽

```
1 #include stdout.h
```

not used

Note 7. The header contains a link to chunk 2b which is the next chunklet of this named chunk.

Main Body

The include statement is followed by our main function that we defined in 1d.

2b ⟨./hello-world.c[2]() ⇑2a, lang=cpp⟩ +≡ △2a 2c▽

```
2 ⟨main 1d⟩
```

Note 8. This time the header also contains a link to chunk ? which is the previous chunklet of this named chunk.

Closing remarks

And a final good-bye comment.

2c ⟨./hello-world.c[3]() ⇑2a, lang=cpp⟩ +≡ △2b

```
3 /* thats the end, folks */
```

Note 9. Like the first chunks we looked at, there are no further chunklets with the same name and so there is no link to the next chunklet shown, although there is a link to the previous chunklet.

5 Updating the Document

Because of the amount of page-referencing going on to calculate the chunklet references, it can sometimes require the document to be updated three or more times to fix the links and references.

1. The first time, a chunk has to discover which page it is one.
2. The second time all the chunks can be given the correct id based on the page.
3. The third time, all references to a chunk can be updated to the correct id.
4. A forth and additional times may be required if the page-breaking was changed as a result of a change in label size — although this is only likely to occur if a reference occured in the text of the document for the references in the header have enough space so as not to affect layout.

6 Extracting the source

The file `./hello-world.c` can be extracted directly with:

```
texmacs -s -c hello-world.tm hello-world.txt -q &&
fangle -R./hello-world.c hello-world.txt > hello-world.c
```

Although this is a little verbose, it shows how things work. There is a Makefile.inc project that allows you to do all that sort of thing automatically.

Code Index