Chapter 1

Creating documentation

This document provides an overview of how to prepare documentation for inclusion within `oomph-lib`.

1.1 Setting up the documentation

1.1.1 Makefiles

1. Within an appropriate subdirectory of the `doc` directory, create a directory for your documentation, e.g.
   ```
   cd doc/axisym_navier_stokes/
   mkdir spin_up
   ```
2. Add this new directory to the parent directory's `Makefile.am`
3. Add the entry `doc/axisym_navier_stokes/spin_up` to the file `config/configure.ac_scripts/doc.dir_list`
4. Create a new `*.txt` file with the same name as the newly-created directory, e.g.
   ```
   touch spin_up.txt
   ```
   This is the "source" file from which the documentation will be generated.
5. From an existing documentation directory, copy across the following files into the newly created directory:
   - `Makefile.am`
   - `Doxyfile`
6. In `Makefile.am` after `"docfile =`", add the stem of the `*.txt` file created in step 4, e.g.
   ```
   docfile = spin_up
   ```
7. In `Doxyfile` update the (relative) path to the demo-driver directory. Two entries must be updated, one following `"INPUT"` and the other following `"EXAMPLE_PATH"`. To find these, search for `"../.."`.
8. Return to `oomph-lib's` top-level directory and re-run `autogen.sh:`
   ```
   ./autogen.sh
   ```

1.1.2 Figures

If your documentation is to contain figures or animations, the following subdirectories must be created within your documentation directory:

- `figures`
  - This must contain all figures used in the documentation in both `*.gif` and `*.eps` formats (`*.gif` is used for the `html` documentation whilst `*.eps` is used for `LaTeX`).
- `non_distfigures`
  - This contains any additional files (etc.) used to create or maintain the documentation. It is good practise to always keep any `*.lay` and `*.lpk` files that were used to create figures, along with any macro (`*.mcr`) files.

1.2 Writing the documentation

The "source" of the documentation is contained in the *.txt file (a mixture of html/doxygen markup). The title must follow the tag \mainpage and be all on one line, e.g.

```
\mainpage Demo problem: Spin-up of a viscous fluid
```

The main body of the text just follows as in a LaTeX document, with line spacings indicating paragraph breaks.

1.2.1 HTML

Any standard html tags can be used (for example, <hr> inserts a horizontal line). Hyperlinks are inserted in the following way:

```
This is illustrated in an <A HREF="../figures/my_movie.avi">animation</A>.
```

Note that the "../" is present in the path above because during the build process the documentation is created in subdirectories of the directory in which the source file is located. See the section Generating the documentation below for more details.

To link to the documentation of another demo driver, poisson/one_d_poisson for example, hyperlink to the index.html file within the html directory of that demo driver's documentation. Note that because the processed (html) version of the documentation you are working on will live in its own html subdirectory, it is necessary to go up three directories in order to be in oomph-lib's doc directory.

```
...see <A HREF="../../../poisson/one_d_poisson/html/index.html">the
Poisson tutorial</A>.
```

1.2.2 Sections

Sections are created as follows:

```
\section section_label This is the title of my section
```

Likewise, subsections are created in the following way:

```
\subsection subsection_label This is the title of my subsection
```

Any LaTeX section types can be used in this way. To link to a section within a document, use the syntax \ref section_label, as in this example:

```
...can be found in the section \ref theory below.
```

1.2.3 Equations

Equations are generated as in LaTeX except \f must be added before $ or [,].

- E.g. inline maths:
  ```
  ...is given by \f$ \sin(x) \f$
  ```
- E.g. equation environment:
  ```
  \f[
  \sin(x)
  \f]
  ```

To label equations, the normal LaTeX system cannot be used. Instead, mark an equation with the tag @E[LABEL]@ and refer to it later using @R[LABEL]@, e.g.

```
\f[
\nabla \cdot \mathbf{u} = 0 \ \ \ \ \ \ \ \ \ \ @E[eqn:cont]@
\f]
```

and then later on...

```
...is given by equation (@R[eqn:cont]@).
```

1.2.4 Lists

To create bullet point lists, precede each item with a -, e.g.

```
- First item
- Second item
```

To create enumerated lists, precede each item with a -#, e.g.

```
-# First item
-# Second item
```

1.2.5 Figures

A figure with the filename `my_figure.*` is inserted in the following way:

```
@I w 0.75\textwidth my_figure "This is the caption. "
```

Note the space between the last character in the caption and the quotation marks. Like the equation labelling, this line is processed by the `txt2h.sh` script (which is run automatically as part of the `make` process) and replaced with the necessary commands that tell `doxygen` to use the `*.gif` files for the html documentation and the `*.eps` files for the `LaTeX` documentation.

1.2.6 Code

To insert single words of code into prose, precede the word with a `\c`, e.g.

```
The function \c FiniteElement::output(...) is used to...
```

To include blocks of code such as the one immediately above this line of text, use the `\code` environment, e.g.

```
\code
for(unsigned i=0; i<10; i++) { cout << This is my sample code << endl; }
\endcode
```

To include sections of the demo code which you are documenting, e.g. the main function of `spin_up.cc`, use the following syntax:

```
\dontinclude spin_up.cc \skipline start_of_main
\until end of main
```

This only works if `start_of_main` exists somewhere in `spin_up.cc` file, but any word(s) can be used as a start/endpoint. However, **do not use dashes as targets** because more recent versions of doxygen get very confused by this, so **don't** do

```
\skipline ----
```

say.

1.2.7 Miscellaneous

- To tell `doxygen` to ignore everything in the source file below a certain point, denote this point with `@@END@@`.
- To tell `doxygen` that a certain section of the source file is only to be included in the `html` version of the documentation and omitted in the `LaTeX` version, enclose this section within `\htmlonly and \endhtmlonly` tags. CAREFUL: With recent version of doxygen, this has caused problems with certain commands not being interpreted correctly. Best not to use this... The following item is a work-around:
- Add the variable `suppress_latex_in_this_directory` to the Makefile.am and set it to 1 to bypass the generation of latex-based documentation for a specific directory (which may contain difficult to render tables etc. and therefore cause latex to hang...). Here's an example of a Makefile.am from the directory `doc/order_of_action_functions`:

```
suppress_latex_in_this_directory=1
include $(top_srcdir)/config/makefile_templates/doc
docfile = order_of_action_functions
```

- To tell `doxygen` that a certain section of the source file is only to be included in the `LaTeX` version of the documentation and omitted in the `html` version, enclose this section within `\latexonly and \endlatexonly` tags.

1.3 Generating the documentation

Once the source file has been written, simply type `make` in the documentation directory to build the `html` and `LaTeX` versions, e.g.

```
cd doc/axisym_navier_stokes/spin_up
make
```

Two subdirectories, `html` and `latex`, are now created containing the two versions of the documentation. A `*.pdf` file of the `LaTeX` version is also placed in the current directory.

1.4 PDF file

A pdf version of this document is available.