coqweb: a WEB-like tool for Coq

Jean-Christophe Filliâtre
http://www.lri.fr/~filliatr/coqweb

Contents

1 Introduction **1**

2 Principles **1**

3 Hints to get a pretty document **3**

4 Usage **3**

5 The coqweb LaTeX style file **5**

6 FAQ **6**

1 Introduction

coqweb is a "literate programming" tool for the proof assistant Coq, similar to ocamlweb, a literate programming tool for Objective Caml (see www.lri.fr/~filliatr/ocamlweb). These two tools are very close and the principles are exactly the same. However, this document does not assume any knowledge of ocamlweb.

The task of coqweb is to:

1. produce a nice document from the Coq sources, readable for a human and not only for the proof assistant;
2. generate a global index, to help the user navigating in his own (or third-party) sources.

2 Principles

Documentation is inserted into Coq files as *comments*. Thus your files may compile as usual, whether you use coqweb or not. coqweb presupposes that the given Coq files are well-formed (at least lexically, but also syntactically if you want the cross references to be correct). coqweb understands each Coq file as a sequence of paragraphs, each paragraph being either a piece of vernacular or a piece of documentation. Documentation starts and ends with Coq

comment's delimiters (* and *), and is a regular LaTeX text. Vernacular starts with any characters sequence other than (* and ends with an empty line.

Two styles are available for the final document:

- a WEB style, where the code and documentation are displayed in *sections*, numbered consecutively starting from 1. A new section is started at the beginning of each file. You can also start a new section manually (see the paragraph below about controls);
- a LaTeX style, where you structure your document as you like, using the usual sectioning commands of LaTeX.

Escapings: vernacular inside documentation and vice versa. The first feature you will require is the ability to quote vernacular inside documentation and conversely. The latter, namely documentation inside code, is simply obtained by normal Coq comments filled in with a LaTeX contents. The former is obtained by quoting vernacular between the delimiters [and]. Square brackets may be nested, the inner ones being understood as being part of the quoted code (thus you can quote a term like $[x : T]u$ by writing [[x:T]u]). Inside quotations, the code is pretty-printed in the same way as it is in code parts.

Controls. In addition to the default behavior of coqweb, you can control it through a small set of commands, which are comments of a particular shape. These commands are the following:

(*s

Starts a new section. (Meaningless in LaTeX style.)

(*i ... i*)

Ignores all the text between those two delimiters. Such "comments" cannot be nested but any Coq vernacular and comments may appear between them, included nested Coq comments. You can use those delimiters to enclose a comment that shouldn't appear in the document, but also to hide some Coq vernacular, putting it between (*i*) and (*i*) in such a way that your vernacular is still interpreted but is ignored by coqweb. (Notice that if you use the delimiters (*i and i*) directly, the text enclosed is commented for both Coq and coqweb.)

(*c

Tells coqweb that this is a real Coq comment an not a documentation text. Mainly useful is you want to insert comments after empty lines. However, this is not the spirit of coqweb, and you are encouraged to put the documentation in separate LaTeX paragraphs and not in traditional Coq comments.

(*r

Justifies an end-of-line comment on the right margin.

(*p ... *)

Insert some material in the LaTeX preamble. See also command line option -p.

Pretty-printing. coqweb uses different faces for identifiers and keywords, and use mathematical symbols for some Coq tokens. Here are the correspondences.

`->`	$\rightarrow$	`<-`	$\leftarrow$	`*`	$\times$
`<=`	$\leq$	`>=`	$\geq$	`=>`	$\Rightarrow$
`<>`	$\neq$	`<->`	$\leftrightarrow$	`|-`	$\vdash$
`\/`	$\vee$	`/\`	$\wedge$	`~`	$\neg$

Characters strings are pretty-printed in fixed-width font, and spaces are explicitly shown as the symbol ␣, as in `"a␣constant␣string"`.

3 Hints to get a pretty document

coqweb is rather robust, in the sense that it will always succeed in producing a document, as soon as the Coq files are lexically correct. However, you may find the result rather ugly if so were your source files. Here are some style hints to get a pretty document.

First, coqweb is not a real code pretty-printer: it does not indent your code, and it does not even cut your code lines when they are too long. The code will always appear formatted as it is in the source files. The indentation at the beggining of each line is output as a proportional space (tabulations are correctly translated). In particular, if you want your Cases construct to be aligned, you have to put a '|' also in front of the first pattern. Here is the difference:

```
Cases e with
     O => ...
   | (S p) => ...
end
```

```
Cases e with
   | O => ...
   | (S p) => ...
end
```

4 Usage

coqweb is invoked on a shell command line as follows:

`coqweb` *< options and files >*

Any command line argument which is not an option is considered to be a file (even if it starts with a `-`). Coq files are identified by the suffixes `.v` and `.g` and LaTeX files by the suffix `.tex`. The latter will be copied 'as is' in the final document. The order of files on the command line is kept in the final document.

Command line options

`-o` ***file***, `--output` ***file***

Redirects the output into the file '*file*'.

`--noweb`

In that case, there are no sections à la WEB, and the user structurates his document directly through LaTeX commands, like for instance `\section`, `\subsection`, etc. There is still an index, in the usual LaTeX sense. (Therefore, if you have introduced LaTeX sections and subsections, their numbers will appear in the index.)

`-s , --short`

Do not insert titles for the files. The default behavior is to insert a title like "Module Foo" for each file.

`--no-index`

Do not output the index.

`--extern-defs`

Keeps the external definitions in the index i.e. the identifiers which are not defined in any part of the code. (The default behavior is to suppress them from the index, even if they are used somewhere in the code.)

`--header`

Does not skip the header of Coq files. The default behavior is to skip them, since there are usually made of copyright and license informations, which you do not want to see in the final document. Headers are identified as comments right at the beginning of the Coq file, and are stopped by any character other then a space outside a comment or by an empty line.

`--no-preamble`

Suppresses the header and trailer of the final document. Thus, you can insert the resulting document into a larger one.

`-p` ***string*****,** `--preamble` ***string***

Insert some material in the LaTeX preamble, right before `\begin{document}`. See also the control `(*p`.

`--latex-option` ***option***

Passes the given option the LaTeX package `coqweb.sty`.

`--vernac` ***file*****,** `--tex` ***file***

Considers the file '*file*' respectively as a `.v` (or `.g`) file or a `.tex` file.

`--files` ***file***

Read file names to process in file '*file*' as if they were given on the command line. Useful for program sources splitted in several directories. See FAQ.

`-q, --quiet`

Be quiet. Do not print anything on standard error output except errors.

`-h, --help`

Gives a short summary of the options and exits.

`-v, --version`

Prints the version and exits.

5 The `coqweb` LaTeX style file

In case you choose to produce a document without the default LaTeX preamble (by using option `--no-preamble`), then you must insert into your own preamble the command

`\usepackage[`*options*`]{coqweb}`

Alternatively, you may also pass these options with the `--latex-options` option of the `coqweb` command.

The options that you may pass to the package are the following:

`noweb`

Tells `coqweb.sty` that the document was generated with the `--noweb` option so that no WEB sections are used, and the index was generated by referencing the LaTeX sections, subsections, etc.

`novisiblespaces`

By default, spaces in strings of Coq source parts are output as ␣. They will be output as real spaces if you select this option.

`bypages`

When sectioning is done by LaTeX sectioning commands, this option asks that the index must refer to page numbers instead of section numbers. In WEB sectioning style, this option has no effect.

Additionally, you may alter the rendering of the document by redefining some macros:

`coqwkw, coqwupperid, coqwlowerid`

The one-argument macros for typesetting keywords, uppercase identifiers and lowercase identifiers respectively. Defaults are sans-serif for keywords and italic for identifiers.

For example, if you would like a slanted font for keywords, you may insert

```
\renewcommand{\coqwkw}[1]{\textsl{#1}}
```

anywhere between `\usepackage{coqweb}` and `\begin{document}`.

`coqwmodule`

One-argument macro for typesetting the title of a `.v` file. Default is

```
\newcommand{\coqwmodule}[1]{\section*{Module #1}}
```

and you may redefine it using `\renewcommand`.

6 FAQ

1. **What about an HTML output?**

   Use hevea, the LaTeX to HTML translator written by Luc Maranget (freely available at `http://pauillac.inria.fr/hevea/`), with the following command-line:

   `hevea coqweb.sty` *file.tex*

   where *file.tex* is the document produced by coqweb. The package `coqweb.sty` contains the necessary support for hevea.

2. **How can I customize the appearance of the final document?**

   You can redefine some of the LaTeX macros of the `coqweb.sty` style file. See Section 5. We do not recommend to modify the `coqweb.sty` file directly, since you would not be able to update easily to future versions. You should rather redefine the macros using `\renewcommand`. If you want to customize other parameters that are not currently customizable, please contact the developers.

3. **How can I insert 'usepackage' commands, or whatever else, in the LaTeX preamble?**

   Use the option `-p` or the corresponding control `(*p`.

   If you really want a different LaTeX preamble, for instance to use a LaTeX class other than `article`, then use the option `--no-preamble` and catenate the result with your own header and trailer (or use a `\input` or `\include` command to insert the file generated by coqweb in your main LaTeX file.)

4. **How can I use square brackets in LaTeX, since they are reserved characters for coqweb?**

   There is no escape sequence for the moment. But there is an easy solution to this problem: define macros for your LaTeX notations involving square brackets and put them in the preamble or in a `.tex` file that you will provide on the coqweb command line. Then, there is no longer square brackets in your source comments, but only calls to these macros.

5. **I would like my Cases right-hand sides to be aligned. Is it possible?**

   No, it is not, since coqweb uses proportional fonts. But you can still align your pattern-matching in your source files, since coqweb converts multiple spaces into single ones.