

# erl2pdf: Literal Erlang Programming

Ulf Wiger <ulf@wiger.net>

March 16, 2009

Copyright (c) 2008 Ulf Wiger, John Hughes<sup>1</sup>

```
-module(erl2pdf).  
  
-export([file/1, file/2]).
```

## 1 Introduction

This module converts an Erlang source file to pdf. It uses latex as an intermediate format, and removes the temporary files afterwards.

The idea of ‘literal Erlang programming’ is that the source and comments should read as a good paper. Unlike XML markup, Latex markup is also fairly unobtrusive when reading the source directly.

```
file(F) ->  
    file(F, []).  
  
file(F, Opts) ->  
    case file:read_file(F) of  
        {ok, Bin} ->  
            Tmp_dir = tmp_dir(),  
            Res =  
                try  
                    Latex_options = proplists:get_value(latex, Opts, []),  
                    case erl2latex:file(  
                        F, [{outdir, Tmp_dir} | Latex_options]) of  
                            {ok, Latex} ->  
                                make_pdf(Latex, Tmp_dir, Opts);
```

---

<sup>1</sup>The MIT License

Copyright (c) 2008 Ulf Wiger <ulf@wiger.net>, John Hughes <john.hughes@quviq.com>

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

```

        Err ->
            Err
        end
    catch
        error:R ->
            {error, R}
    after
        case proplists:get_value(keep_temp,Opts,false) of
            true ->
                io:fwrite("Intermediate files in ~s~n", [Tmp_dir]);
            false ->
                remove_dir(Tmp_dir)
        end
    end,
    Res;
    Open_err ->
        Open_err
end.

```

## 2 Calling pdflatex

We step into the temporary directory, and call `pdflatex`, which outputs a huge amount of diagnostics. It also sets an exit code on error, but for various reasons, exit codes are difficult to catch using `os:cmd/1`. We try to detect errors when moving the generated pdf to its intended location; if this fails, we assume that `pdflatex` failed and print the result from the `os:cmd()` call. Ok, so this is a dirty hack. See it as an opportunity to improve the code.

```

make_pdf(Latex, TmpDir, Opts) ->
    Cmd = ["cd ", TmpDir, "; echo \"in ${PWD}\"; ls; ",
           "pdflatex ", filename:basename(Latex)],

    OutDir = proplists:get_value(outdir,Opts,"."),

    Res = os:cmd(Cmd),

    case check_res(Res) of
        ok ->
            PdfFile = pdf_file(Latex),
            case file:rename(PdfFile,
                            filename:join(OutDir,
                                           filename:basename(PdfFile))) of
                ok ->
                    ok;
                {error,_} ->
                    io:fwrite("~s~n", [Res]),
                    error
            end;
        {error,Err} ->
            io:fwrite("~s~n", [Err]),
            error
    end.

check_res(Res) ->
    Lines = string:tokens(Res,"\\n"),

```

```

case lists:dropwhile(fun("!" ++ _) ->
                        false;
                      (_,) ->
                        true
                    end, Lines) of
[] ->
    ok;
[L|Ls] ->
    {error, lists:concat([L|[$\n|Lx] || Lx <- Ls])}
end.

```

### 3 Helper Functions

The pdf file name differs from the latex file name only in the extension.

```

pdf_file(Latex) ->
    "xet." ++ Rev = lists:reverse(Latex),
    lists:reverse("fdp." ++ Rev).

```

Generate a temporary directory.

```

tmp_dir() ->
    {A,B,C} = erlang:now(),
    D = lists:flatten([?MODULE_STRING,
                       ["." ++ integer_to_list(I) || I <- [A,B,C]]]),
    ok = filelib:ensure_dir(D),
    ok = file:make_dir(D),
    D.

```

Remove the temporary directory

```

remove_dir(D) ->
    remove_files(D),
    ok = file:del_dir(D).

remove_files(D) ->
    {ok,Fs} = file:list_dir(D),
    [ok = file:delete(filename:join(D,F)) || F <- Fs].

```