

```
3|  
4| use File::Temp;  
5| use Semi::Literate;
```

## Weave a semi-literate program into Text, Markdown, etc. format

```

6| sub MAIN($input-file,
7|         Bool :l(:$line-numbers) = True;
8|         Str :f(:$format) is copy = 'markdown';
9|         Str :o(:$output-file);
10|        Bool :v(:$verbose) = True;
11|    ) {
12|    my Str $extension;
13|    my Str @options;
14|    my Bool $no-output-file = False;
15|
16|    note "Input Format => $format" if $verbose;
17|    $format .= trim;
18|    given $format {
19|        when /:i ^ markdown | md $ / {
20|            $format = 'Markdown2';
21|            $extension = 'md';
22|        };
23|        when /:i ^ [[plain][\-|_]]? t[e]?xt $ / {
24|            $format = 'Text';
25|            $extension = 'txt';
26|        }
27|        when /:i ^ [s]?htm[l]? $/ {
28|            $format = 'HTML2';
29|            $extension = 'html';
30|        }
31|        when /:i ^ pdf $ / {
32|            $format = 'PDF';
33|            $extension = '.pdf';
34|            @options = "--save-as=$output-file" if $output-file;
35|            $no-output-file = True;
36|        }
37|        when /:i ^ pdf[\-|_]?lite $ / {
38|            $format = 'PDF::Lite';
39|            $extension = '.pdf';
40|            @options = "--save-as=$output-file" if $output-file;
41|            $no-output-file = True;
42|        }
43|        when /:i ^ pod 6? $/ {
44|            $format = 'Pod6';
45|            $extension = '.rakudoc';
46|        }
47|
48|        default {
49|            $extension = $format;
50|        }
51|    }
52|
53|    my Str $woven = weave($input-file, :$line-numbers);
54|
55|    my $output-file-handle = $output-file ??
56|                            open(:w, $output-file) !!
57|                            $*OUT
58|                            unless $no-output-file;

```

```
59|
60|     if $format eq 'Pod6' {
61|         $output-file-handle.spurt: $woven;
62|         return;
63|     }
64|
65|     note "Weave Format => $format" if $verbose;
66|     my Str $f = "Pod::To::$format";
67|     try require ::($f);
68|     if ::($f) ~~ Failure {
69|         die "$format is not a supported output format"
70|     }
71|
72|     my ($pod-file, $fh) = tempfile(suffix => '.rakudoc', :!unlink);
73|     note "Temp file: $pod-file" if $verbose;
74|
75|     $pod-file.IO.spurt: $woven;
76|
77|     run $*EXECUTABLE,
78|         "--doc=$format",
79|         $pod-file,
80|         @options,
81|         :out($output-file-handle);
82|
83| }
84|
```