Nancy, the lazy web site builder
User's guide

Reuben Thomas

8th January 2008

1 Introduction

Nancy is a simple web site builder that glues together HTML and other fragments to make pages, and allows fragments to be specialised for particular pages. Fragments can be held in files or generated programmatically.

Nancy is a command-line tool, typically used interactively from a POSIX-like shell.

2 Invocation

Nancy takes three (or optionally four) arguments:

```
nancy SOURCE DESTINATION TEMPLATE [BRANCH]
```

where `SOURCE` is the directory that contains the source tree, `DESTINATION` is the directory to which the resulting HTML pages will be written, `TEMPLATE` is the name of the template file, and the optional `BRANCH` gives the sub-directory of `SOURCE` to process (if it is omitted, the entire source tree is processed).

If you supply the optional flag `-list-files`, or `-l`, the files read by nancy will be listed on standard error.

3 Operation

Nancy produces the finished pages according to the following algorithm:

1. For each leaf directory of the source tree, start from that directory.
2. Set the initial text to `$include{TEMPLATE}`.
3. Repeatedly scan the text for a command and replace it by its output, until no more commands are found.
4. Write out the resultant text to a file: for each directory `SOURCE/LEAF_PATH` the output file is `DESTINATION/LEAF_PATH.html`.

The reason that only leaf directories correspond to pages is to ensure that every page can be specialised without affecting any other page. By convention, every non-leaf directory has an `index` sub-directory, so that there are no "missing" URLs in the resulting site. Nancy warns if an `index` directory is absent.

A command takes the form

```
$COMMAND{ARGUMENT, ...}
```

Nancy recognises these commands:

`$include{FILE}` Replace the command with the contents of the given file.

`$root{}` Replace the command with the relative URL from the page currently being constructed to the root of the site. This means that every link in a site can written relative to the current page, either explicitly (which makes sense for pages related to the current page), or implicitly as `<a href="root{}/path/to/page.html">`. Hence the site's base URL can be changed without needing to change any intra-site links.

`$run{PROGRAM[, ARGUMENT, ...]}` Replace the command with the output of the given program. The command is run with the full path of the current fragment as the first argument, with any further arguments given in the command after that.

Only one guarantee is made about the order in which commands are processed: if one command is nested inside another, the inner command will be processed first. (Other than that, the order can only matter for —`$run` commands; if you use them, you have to deal with this potential pitfall.)

To find the file `FILE_PATH` specified by an `$include` or `$run` command, nancy proceeds thus:

1. Look in `DIRECTORY/LEAF_PATH/FILE_PATH`.
2. If the file is not found, remove the final directory from `LEAF_PATH` and try again, until `LEAF_PATH` is empty.
3. Finally, try looking in `DIRECTORY/FILE_PATH`.

So, for example, if `DIRECTORY` is `/dir`, `LEAF_PATH` is `foo/bar/baz` and nancy is trying to find `file.html`, it will try the following directories, in order:

1. `/dir/foo/bar/baz/file.html`
2. `/dir/foo/bar/file.html`
3. `/dir/foo/file.html`
4. `/dir/file.html`

4 Example

Suppose a web site with the following page design, from top to bottom: logo, navigation menu, breadcrumb trail, page body.

Most of the elements are the same on each page, but the breadcrumb trail has to show the canonical path to each page, and the logo is bigger on the home page.

Suppose further that the web site has the following structure, where each line corresponds to a page:

- Home page
- People
  - Jo Bloggs
  - Hilary Pilary
  - ...
- Places
  - Vladivostok
  - Timbuktu
  - ...

The basic page template looks something like this:

```
<html>
  <link href="style.css" rel="stylesheet" type="text/css">
  <title>$include{title}</title>
  <body>
    <div class="logo">$include{logo.html}</div>
    <div class="menu">$include{menu.html}</div>
    <div class="breadcrumb">$include{breadcrumb.html}</div>
    <div class="main">$include{main.html}</div>
  </body>
</html>
```

Making the menu an included file is not strictly necessary, but, as in programming, makes the HTML fragments easier to read. The pages will be laid out as follows:

- /
  - index.html
  - people/
    - index.html
    - jo_bloggs.html
    - hilary_pilary.html
  - places/

        * index.html
        * vladivostok.html
        * timbuktu.html

The corresponding source files will be laid out as follows. This may look a little confusing at first, but note the similarity to the HTML pages, and hold on for the explanation!

- source/
  - template.html (the template shown above)
  - menu.html
  - logo.html
  - breadcrumb.html
  - index/
    * main.html
    * logo.html
  - people/
    * breadcrumb.html
    * index/
      · main.html
    * jo_bloggs/
      · main.html
    * hilary_pilary/
      · main.html
  - places/
    * breadcrumb.html
    * index/
      · main.html
    * vladivostok/
      · main.html
    * timbuktu/
      · main.html

We could have used a different file suffix for page fragments, but using `.html` is not too confusing, and means that editors and other tools that might depend on the file suffix to treat the file properly don't need special attention.

Note that there is only one menu fragment (the main menu is the same for every page), while each section has its own breadcrumb trail (`breadcrumb.html`), and each page has its own content (`main.html`).

To build the site, nancy is invoked as:

```
nancy source template.html dest
```

Now consider how nancy builds the page whose URL is `vladivostok.html`. According to the rules given in Section 3, nancy will look first for files in `source/places/vladivostok`, then in `source/places`, and finally in `source`. Hence, the actual list of files used to assemble the page is:

- source/template.html
- source/logo.html
- source/menu.html
- source/places/breadcrumb.html
- source/places/vladivostok/main.html

For the site's index page, the file index/logo.html will be used for the logo fragment, which can refer to the larger graphic desired.

This scheme, though simple, is surprisingly flexible; this simple example has covered all the standard techniques for nancy's use.