Nancy, the lazy web site maker
User's guide

Reuben Thomas

4th November 2011

1 Introduction

Nancy is a simple web site maker that glues together HTML fragments and other files to make pages, and finds all the pages and files that make up a site. Files can be generated programmatically, and specialized for particular pages.
Nancy is a CGI program which generates pages on the fly.

2 Invocation

Nancy is configured via environment variables:

`NANCY_SOURCES` a list of directories that contain the source (separated by the system directory separator, usually a colon)

`NANCY_TEMPLATE` the template name, which defaults to `template`

`NANCY_HOME` gives the home page of the site, which defaults to `index.html`

`NANCY_LIST_FILES` List on standard error the files used to make each page.

3 Operation

Nancy produces the site according to the following algorithm:

1. Start with the `HOME`.

2. For the current object, with base name `foo.bar`:
   a) If it is a file, copy it to the output.
   b) Otherwise, assuming it is a directory:
      i. Set the initial text to `$include{TEMPLATE.bar}`.
      ii. Repeatedly scan the text for a command and replace it by its output, until no more commands are found.

iii. Find all the `href` and `src` links in the text, and process each as the current object.

iv. Write the resultant text to a file: for each directory `SOURCE/PATH` the output file is `DESTINATION/PATH`.

In general, only leaf directories, that is, directories that only contain files, should correspond to pages: this is to ensure that every page can be specialised without affecting any other page. It is advisable to ensure that every non-leaf directory has an `index.html` sub-directory (or some other valid index page name), so that there are no URLs in the resulting site that do not correspond to a page.

A command takes the form

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

Nancy recognises these commands:

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

`$root{}` Replace the command with the relative URL to the root of the site from the page under construction. This means that every link in a site can be written relative to the current page, either explicitly (which is a good way to link to pages related to the current page, as such links do not need to be rewritten if the related pages are moved together within the site), or implicitly as `<a href="$root{}/path/to/page.html">`. Hence the site's base URL can be changed without needing to change any intra-site links.

`$run{FILE[, ARGUMENT, ...]}` Replace the command with the output of the given file evaluated as a Perl expression, which is expected to produce a subroutine, which is then called with the given arguments, followed by the path of the page currently being expanded, represented as a reference to an array of path components, left-to-right, and a list of the source tree roots, each represented in the same way, one extra argument per root.

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. (The order only matters for `$run` commands; if you nest them, you have to deal with this potential pitfall.)

To find the file `FILE` specified by a `$include` or `$run` command, Nancy proceeds thus:

1. See whether `SOURCE/PAGE_PATH/FILE` exists in any source tree, trying the trees in left-to-right order.

2. If the file is not found, remove the last directory from `PAGE_PATH` and try again, until `PAGE_PATH` is empty.

3. Finally, try looking in `SOURCE/FILE`.

For example, if `SOURCES` is `/dir` and `FILE` 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 has 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, which is the default `index.html`.

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 makes the template 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.html/
    * main.html
    * logo.html
  - people/
    * breadcrumb.html
    * index.html/
      · main.html
    * jo_bloggs.html/
      · main.html
    * hilary_pilary.html/
      · main.html
  - places/
    * breadcrumb.html
    * index.html/
      · main.html
    * vladivostok.html/
      · main.html

∗ `timbuktu.html/`

· `main.html`

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`).

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.html`, 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.html/main.html`

For the site's index page, the file `index.html/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.