Project Web

This file explains the structure of the gawkextlib public web site, and give instructions about how to generate individual web pages for each extension almost automatically.

Web structure

The top directory contains the global index page and some additional files directly linked by it. This page also has links to the web page of each available extension.

Each individual extension has a separate directory with a single web page that describes it and has links to its documentation. The linked documents are also held in this directory. The name of each directory is the short name of the extension.

```
topdir/
  |- index.html
  |- gawkextlib.css
  |- ...
  |- abort/
  |    |- abort.html
  |    |- abort.3am.html
  |    +- abort.3am.pdf
  |- errno/
  |    |- errno.html
  |    |- ...
  |-...
```

Document types

The information used to construct the web is taken from files stored in the gawkextlib repository. Several types of files can be processed in order to build the web. They are:

- Plain text files (no suffix)
- Man page files (.3am)
- Texinfo files (.texi)
- Markdown files (.md)
- awk source files (.awk)

If the whole file has to be put in the web as a link target, it is converted to both HTML and PDF formats. To be easily readable.

The contents of a plain text file can also be directly included as part of the appropriate web page.

Web contents specification

An extension web page will contains stuff from the extension source files. The page is simply a means to access the relevant information to be directly readable.

Each extension source tree should have a `webTOC` plain text file that will be use to generate the web page automatically. For instance, the specification of the redis web page could be:

```
gawk-redis: Redis extension of GNU awk

Description
<- README

Authors
<- AUTHORS
```

```
Documentation
gawk-redis manual -> doc/README.md
--> awkRedis.png

External links
Redis server *-> http://redis.io/
hiredis C client for Redis *-> https://github.com/redis/hiredis
```

The syntax of this file is expected to be intuitive. There are several line types:

`gawk-redis: Redis extension of GNU awk`

The first line will be taken as the top level header of the page, and converted to an `<h1>` tag.

`Description`

Lines without special marks are handled as section headers, and converted to `<h2>` tags.

`<- README`

Lines starting with a `<-` mark specify text files whose contents will be directly included in the web page, as a series of paragraphs.

`gawk-redis manual -> doc/README.md`

Lines with a `->` mark specify links to local document files. Each link will generate an tag. In addition, consecutive links will be structured as bulleted list items.

The referenced file will be converted to both HTML and PDF formats, and put into the extension web directory.

`--> awkRedis.png`

Lines with a `-->` mark denote additional files that will be also copied to the extension web directory. Like image files used by the HTML version of some document.

`Redis server *-> http://redis.io/`

Lines with a `*->` mark will generate links to external web resources.

Generation procedure

There are several scripts than can be used to automate the process. They are located in the root directory of the gawkextlib repository. There is also a `_web` directory that contains a copy of the project web site, as well as several auxiliary stuff. The structure of this machinery is as follows:

```
gawkextlib home\
  |...
  |- _web\
  |    |- index.html
  |    |- gawkextlib.css
  |    |- ...
  |    |- abort/
  |    |    |- abort.html
  |    |    +- ...
  |    |- errno/
  |    |    |- errno.html
  |    |    +- ...
  |    |- ...
  |    |- makeweb-code.sh
  |    |- makeweb-man.sh
```

```
|    |- makeweb-md.sh
|    |- makeweb-texi.sh
|    +- template.html
|- ...
|- makeweb.sh
|- publish.sh
+- webTOC
```

The `makeweb.sh` script can be used to generate the web directory of a given extension. Example:

```
./makeweb.sh  redis
```

This command will process the `webTOC` file in the redis source tree and populate the `_web/redis` directory with the specified files. The generated web page can be tested by opening the `_web/redis/redis.html` file in a web browser.

The `makeweb.sh` command can be used only for the web pages of individual extensions. The home page of the gawkextlib web site must be manually edited. Anyway, the global documentation linked from the home page can be generated by the command:

```
./makeweb.sh .
```

The web page of an extension can also be generated from a released tarball. This is necessary if the source tree has been modified after releasing the tarball and the web page has not been updated yet to match the new release. In that case the tarball must be unpacked into a temporary directory, and then passed to the makeweb command as a second argument. Example:

```
tar -C /tmp -xf gawk-redis-1.7.4.tar.gz
./makeweb.sh redis /tmp/gawk-redis-1.7.4
```

Note: Tarballs released before the web automation era will require to manually add an adequate `webTOC` specification file to the unpacked directory.

Publishing procedure

The `publish.sh` script contains commands to upload the copy of the web in the `_web` directory to the SourceForge host server. The argument to this script is the name of the extension to publish.

```
./publish.sh redis
```

This script can also be used to upload the files in the home page. The argument will be just a dot.

```
./publish.sh .
```

Note: The `publish.sh` script must be invoked by a developer who has been granted publishing capabilities ("Release Technician" or "with shell permissions" in the SourceForge jargon).

Text file inclusion

The contents of a plain text file to be included in the web page is formatted by applying simple rules:

- Blank lines delimit paragraphs. They generate `<p>` tags.
- Indented blocks means preformatted text. They generate `<pre>` tags.
- Two or more contiguous blank lines will end the included text.

The last rule aims to keep the extension web page relatively short. If a text file to be included is too long, then only the first part should appear in the web page.

Use of Markdown

There are several variants of the Markdown notation. The generator scripts will use the 'pandoc' tool to convert Markdown to HTML and PDF. So the document files of this format should conform to the pandoc's Markdown syntax, described in:

> https://pandoc.org/MANUAL.html#pandocs-markdown

Besides that, files with names ended in `.md` are automatically indentified as Markdown documents. But also text files with no suffix can also be processed as Markdown ones.

To achieve this the file name given in a webTOC specification can add a fake `.md` suffix to the real file name. In that case the generator scripts will seach first for a file with the `.md` suffix and, if not found, will search for a file with the given name but without the `.md` suffix.

Pandoc is used also to convert awk source files. A markdown temporary file is created with the awk code embedded in a fenced code block, and subsequently processed.