NAME

sws — a simple web server

SYNOPSIS

sws [**-dh**] [**-c** *dir*] [**-i** *address*] [**-l** *file*] [**-p** *port*] *dir*

DESCRIPTION

sws is a very simple web server. It behaves quite like you would expect from any regular web server, in that it binds to a given port on the given address and waits for incoming HTTP/1.0 requests. It serves content from the given directory. That is, any requests for documents is resolved relative to this directory (the document root).

OPTIONS

The following options are supported by **sws**:

-c *dir*	Allow execution of CGIs from the given directory. See `CGIs` for details.
-d	Enter debugging mode. That is, do not daemonize, only accept one connection at a time and enable logging to stdout.
-h	Print a short usage summary and exit.
-i *address*	Bind to the given IPv4 or IPv6 address. If not provided, **sws** will listen on all IPv4 and IPv6 addresses on this host.
-l *file*	Log all requests to the given file. See `LOGGING` for details.
-p *port*	Listen on the given port. If not provided, **sws** will listen on port 8080.

DETAILS

sws speaks a crippled dialect of HTTP/1.0: it responds to GET and HEAD requests according to RFC1945, and only supports the If-Modified-Since Request-Header.

In addition, it allows HTTP/1.1 requests to be made, but quietly downgrades the communications to HTTP/1.0.

When a connection is made, **sws** will respond with the appropriate HTTP/1.0 status code and the following headers:

Date	The current timestamp in GMT.
Server	A string identifying this server and version.
Last-Modified	The timestamp in GMT of the file's last modification date.
Content-Type	The correct content-type for the file in question as determined via `magic`(5) patterns.
Content-Length	The size in bytes of the data returned.

If the request type was a GET, then it will subsequently return the data of the requested file. After serving the request, the connection is terminated.

FEATURES

sws supports a number of interesting features:

CGIs	Execution of CGIs as described in `CGIs`.
Directory Indexing	If the request was for a directory and the directory does not contain a file named "index.html", then **sws** will generate a directory index, listing the contents of the directory in alphanumeric order. Files starting with a "." are

ignored.

User Directories — If the request begins with a "/~", then the following string up to the next slash is translated into that user's **sws** directory (e.g., */home/<user>/sws/*).

CGIs

If a URI begins with the string "/cgi-bin", and the **-c** flag was specified, then the remainder of the resource path will be resolved relative to the directory specified using this flag. The resulting file will then be executed and any output generated is returned instead. Execution of CGIs follows the specification in RFC3875.

LOGGING

Per default, **sws** does not do any logging. If explicitly enabled via the **-l** flag, **sws** will log every request in a slight variation of Apache's so-called "common" format: '%a %t "%r" %>s %b'. That is, it will log:

%a	The remote IP address.
%t	The time the request was received (in GMT).
%r	The (quoted) first line of the request.
%>s	The status of the request.
%b	Size of the response in bytes. Ie, "Content-Length".

Example log lines might look like so:

```
155.246.89.8 2019-10-28T12:12:12Z "GET / HTTP/1.0" 200 1406
2001::fe72:3900 2019-10-28T12:12:12Z "GET /file HTTP/1.0" 404 312
```

All lines will be appended to the given file unless **-d** was given, in which case all lines will be printed to stdout.

EXAMPLE INVOCATIONS

In order to test **sws**, you may wish to compare to e.g., bozohttpd(8). For this, let us start bozohttpd(8) in the background (**-b**), listen on port 8080 (**-I** *8080*), enable user translation using the "sws" directory name (**-u** **-p** *sws*), enable CGI execution from the "cgi-bin" directory (**-c** *cgi-bin*) under the document root, and serve data from the directory "docroot":

```
/usr/libexec/httpd -b -I 8080 -X -u -p sws -c cgi-bin docroot
```

The equivalent invocation for **sws** to run side-by-side on port 8081 would then be:

```
sws -p 8081 -c cgi-bin docroot
```

EXIT STATUS

The **sws** utility exits 0 on success, and >0 if an error occurs.

SEE ALSO

RFC1945

HISTORY

A simple http server has been a frequent final project assignment for the class *Advanced Programming in the UNIX Environment* at Stevens Institute of Technology. This variation was first assigned in the Fall 2008 by Jan Schaumann.