Server Side Includes (SSI)

NCSA HTTPd allows users to create documents which provide simple information to clients on the fly. Such information can include the current date, the file's last modification date, and the size or last modification of other files. In its more advanced usage, it can provide a powerful interface to CGI and /bin/sh programs.

SSI Issues

Having the server parse documents is a double edged sword. It can be costly for heavily loaded servers to perform parsing of files while sending them. Further, it can be considered a security risk to have average users executing commands as the server's User[6]. If you disable the exec option, this danger is mitigated, but the performance issue remains. You should consider these items carefully before activating server–side includes on your server.

SSI Setup

First, you should decide which directories you want to allow Includes in. Most likely this will not include users' home directories or directories you do not trust. You should then decide, of the directories you are allowing includes in, which directories are safe enough to use exec in.

For the directories in which you want to fully enable includes, you need to use the Options[7] directive to turn on the option `Includes`. Similarly for the directories you want crippled (no exec) includes, you should use the option `IncludesNOEXEC`. In any directory you want to disable includes, use the Options directive without either option.

Next, you need to tell the server what filename extension you are using for the parsed files. These files, while very similar to HTML, are *not* HTML and are thus not treated the same. Internally, the server uses the magic MIME type `text/x-server-parsed-html` to identify parsed documents. It will then perform a format conversion to change these files into HTML for the client. To tell the server which extension you want to use for parsed files, use the AddType[8] directive. For instance:

1 #Issues
2 #Setup
3 #Convert
4 #Format
5 #Vars
6 ../setup/httpd/User.html
7 ../setup/access/Options.html
8 ../setup/srm/AddType.html

`AddType`[9] `text/x-server-parsed-html .shtml`

This makes any file ending with .shtml a parsed file. Alternatively, if you don't care about the performance hit of having all .html files parsed, you could use:

`AddType`[10] `text/x-server-parsed-html .html`

This would make the server parse all .html files.

Converting your old INC SRV documents to the SSI Format

You should use the program `inc2shtml` in the `support` subdirectory of the HTTPd distribution to translate your documents from HTTPd 1.1 and earlier to the new format. Usage is simple: `inc2shtml file.html > file.shtml`.

The SSI Format

All directives to the server are formatted as SGML comments within the document. This is in case the document should ever find itself in the client's hands unparsed. Each directive has the following format:

```
<!--#command tag1="value1" tag2="value2" -->
```

Each command takes different arguments, most only accept one tag at a time. Here is a breakdown of the commands and their associated tags:

- `config`

  The config directive controls various aspects of the file parsing. There are two valid tags:

  - `errmsg` controls what message is sent back to the client if an error includes while parsing the document. When an error occurs, it is logged in the server's error log[11].
  - `timefmt` gives the server a new format to use when providing dates. This is a string compatible with the `strftime` library call under most versions of UNIX.
  - `sizefmt` determines the formatting to be used when displaying the size of a file. Valid choices are `bytes`, for a formatted byte count (formatted as 1,234,567), or `abbrev` for an abbreviated version displaying the number of kilobytes or megabytes the file occupies.

- `include`

  include will insert the text of a document into the parsed document. Any included file is subject to the usual access control. This command accepts two tags:

  - `virtual` gives a virtual path[12] to a document on the server. You must

9 ../setup/srm/AddType.html
10 ../setup/srm/AddType.html
11 ../setup/httpd/ErrorLog.html
12 ../setup/admin/AccessingFiles.html

access a normal file this way, you cannot access a CGI script in this fashion. You can, however, access another parsed document.

   - `file` gives a pathname relative to the current directory. ../ cannot be used in this pathname, nor can absolute paths be used. As above, you can send other parsed documents, but you cannot send CGI scripts.

- `echo` prints the value of one of the include variables (defined below). Any dates are printed subject to the currently configured `timefmt`. The only valid tag to this command is `var`, whose value is the name of the variable you wish to echo.
- `fsize` prints the size of the specified file. Valid tags are the same as with the `include` command. The resulting format of this command is subject to the `sizefmt` parameter to the `config` command.
- `flastmod` prints the last modification date of the specified file, subject to the formatting preference given by the `timefmt` parameter to `config`. Valid tags are the same as with the `include` command.
- `exec` executes a given shell command or CGI script. It must be activated to be used. Valid tags are:
   - `cmd` will execute the given string using /bin/sh. All of the variables defined below are defined, and can be used in the command.
   - `cgi` will execute the given virtual path to a CGI script and include its output. The server does not perform error checking to make sure your script didn't output horrible things like a GIF, so be careful. It will, however, interpret any URL Location: header and translate it into an HTML anchor.

SSI Environment Variables

A number of variables are made available to parsed documents. In addition to the CGI variable set[13], the following variables are made available:

- `DOCUMENT_NAME`: The current filename.
- `DOCUMENT_URI`: The virtual path to this document (such as /docs/tutorials/foo.shtml).
- `QUERY_STRING_UNESCAPED`: The unescaped version of any search query the client sent, with all shell−special characters escaped with \.
- `DATE_LOCAL`: The current date, local time zone. Subject to the `timefmt` parameter to the `config` command.
- `DATE_GMT`: Same as DATE_LOCAL but in Greenwich mean time.
- `LAST_MODIFIED`: The last modification date of the current document. Subject to `timefmt` like the others.

[13] /cgi/env.html

Return to tutorial index[14]

NCSA HTTPd Development Team / httpd@ncsa.uiuc.edu[15]/ 9−28−95

[14] index.html
[15] ../FeedBack.html