APPENDIX E

Server-Side Includes

Server-Side Includes (SSIs) are provided by `mod_include`, and they provide basic templating capabilities as described in detail in Chapter 6. In particular, they can do the following:

- Add the same content to many pages.
- Insert information from variables to make pages customized and dynamic.
- Perform some functions of CGI programs with conditional SSI commands.

By default, SSI commands have the following syntax:

```
<!--#command attribute="value" attribute="value" ... -->
```

A command is formatted like an HTML comment, so if server-parsed documents aren't enabled, the browser will ignore it, but it'll still be visible in the HTML source. If SSI is correctly configured, the directive will be replaced with its results. In Apache 2, the syntax of the SSI command can be changed with the `SSIStartTag` and `SSIEndTag` directives to create SSI commands that may be seen in GUI editors or are well-formed XML.

In some directives, a distinction is made between specifying a URL and specifying a file path. When a directive requires a file path, Apache treats it as a path relative to the directory containing the document being parsed. It can't contain `../`, and it can't be an absolute path (that is, it can't begin with `/`). This allows SSI commands to include files in the same directory or a subdirectory only. URLs are interpreted either relative to the URL with which the SSI file was called or relative to the server root if they're given as an absolute URL (beginning with a `/`). This allows an SSI command access to any resource that can be accessed by a client.

SSI Directives

Table E-1 describes the SSI directives.

Table E-1. SSI Directives

Directive	Description
SSIErrorMsg	The SSIErrorMsg directive changes the error message displayed when mod_include encounters an error. This directive has the same effect as <--#config errmsg=message -->.
SSIStartTag	This directive changes the string that mod_include looks for to mark the start of an SSI command.
SSIEndTag	This directive changes the string that mod_include looks for to mark the end of an SSI command.
SSITimeFormat	This directive changes the format in which date strings are displayed when echoing DATE environment variables. This directive has the same effect as the <--#config timefmt=formatstring --> element.
SSIUndefinedEcho	This directive changes the string that mod_include displays when a variable isn't set and echoed.
XBitHack	The XBitHack directives control the parsing of ordinary HTML documents. This directive only affects files associated with the MIME type text/html. XBitHack can take on the following values: off: No special treatment of executable files. on: Any file that has the user-execute bit set will be treated as a server-parsed HTML document. full: Same as for on but also tests the group-execute bit. If it's set, then set the Last-Modified header of the response to be the last modified time of the file. If it isn't set, then no last-modified date is sent. Setting this bit allows clients and proxies to cache the result of the request.

SSI Commands

Table E-2 describes the SSI commands.

Table E-2. SSI Commands

Element	Attribute(s)	Notes
config	errmsg	Sets the error message included when the parser encounters badly formatted SSIs to the value given. It defaults to this: [an error occurred while processing this directive]
	sizefmt	Sets the format used by the fsize directive. Acceptable values are either bytes or abbrev. abbrev will give the value in kilobytes or megabytes for files of sufficient size.

(Continued)

Table E-2. SSI Commands (Continued)

Element	Attribute(s)	Notes
	timefmt	Sets the format used by the DATE_GMT, DATE_LOCAL, and LAST_MODIFIED environment variables, as well as the flastmod directive. On Unix, see the man page for strftime on your system to check how the format is specified—the most common syntax is listed in the "Time Format Specifiers" section.
echo	var	The variable to echo as the output of the command.
	encoding	Specifies how Apache should encode special characters contained in the variable before outputting them. If set to none, no encoding will be done. If set to url, then URL encoding (also known as %-encoding) will be performed. At the start of an echo element, the default is set to entity, resulting in entity encoding (which is appropriate in the context of a block-level HTML element, for instance, a paragraph of text). This can be changed by adding an encoding attribute, which will remain in effect until the next encoding attribute is encountered or the element ends, whichever comes first. Note that the encoding attribute must precede the corresponding var attribute to be effective and that only special characters as defined in the ISO-8859-1 character encoding will be encoded. This encoding process may not have the desired result if a different character encoding is in use. Available in Apache 1.3.12 and above; previous versions don't support this attribute.
set	var, value	Sets the value of the environment variable specified by var to the value given by value. var is the name of the variable to set, and value is the value to give a variable.
printenv		Includes a complete list of environment variables and their current values.
exec	cmd	Executes the specified shell command and includes the result if there is one.
	cgi	Executes the specified CGI script (specified by URL) and includes the result. The include variables are available to the command in addition to the usual set of CGI variables. include virtual is preferred. Both forms of the exec directive are disabled if SSI was enabled with options IncludesNoExec.
fsize	file	Includes the size of the specified file, given as a relative path to the directory containing the current document being parsed.
	virtual	Includes the size of the specified file, given as a URL. The size is formatted according to the config sizefmt directive. The value is a (%-encoded) URL-path relative to the current document being parsed.

(Continued)

Table E-2. SSI Commands (Continued)

Element	Attribute(s)	Notes
		If it doesn't begin with a slash (/), then it's taken to be relative to the current document.
flastmod	file	Includes the last modified time of the specified file, given as a relative path.
	virtual	Includes the last modified time of the specified file, given as a URL. The time is formatted according to the config timefmt directive.
include	file	Includes the contents of the specified file, given as a relative path. It can't contain ../, and it can't be an absolute path.
	virtual	Includes the contents of the specified file, given as a URL. The contents returned is the content that Apache would have returned had the file been requested directly—so if the specified file is a CGI script, its output is included. Including executable content is controlled by IncludesNoExec in the same way as exec, with the exception that include virtual can include the result of any CGI script, which can be executed by a client.
if	expr	Allows the construction of conditional sections within an SSI file, based on the evaluation of an expression. The evaluation syntax is explained in the "Condition Evaluation" section.
elif	expr	Allows the construction of conditional sections within an SSI file, based on the evaluation of an expression. The evaluation syntax is explained in the "Condition Evaluation" section.
else		Allows the construction of conditional sections within an SSI file, based on the evaluation of an expression. The evaluation syntax is explained in the "Condition Evaluation" section.
endif		Allows the construction of conditional sections within an SSI file, based on the evaluation of an expression. The evaluation syntax is explained in the "Condition Evaluation" section.

Time Format Specifiers

Specifying a config timefmt directive allows you to specify exactly how you want times to be displayed. The specification is given as a string, which is reproduced literally, except when the parser encounters a % character. The character after the % specifies a time format specifier that's substituted into the time string using the current time and date to derive its value.

Where the time element represents a word—for example, a weekday or the name of a month—the value inserted will be the correct word for the locale of the system (as defined by the environment from which Apache was run).

The allowed elements correspond to those understood by the `strftime` system call (see `man strftime` on any Unix server), as shown in Table E-3.

Table E-3. Time Format Specifiers

Element	Meaning	Example
%a	The abbreviated weekday name	Sun
%A	The full weekday name	Sunday
%b	The abbreviated month name	Apr
%B	The full month name	April
%c	The preferred date and time representation for the current locale	04/11/02 13:04:24
%d	The day of the month as a two-digit number	13
%H	The hour as a two-digit number using the 24-hour clock	23
%I	The hour as a two-digit number using the 12-hour clock	04
%j	The day of the year as a three-digit number	366
%m	The month as a two-digit number	12
%M	The minute as a two-digit number	59
%p	An indicator of whether it's before or after midday, according to the locale	PM
%S	The second as a two-digit number	59
%w	The day of the week as a single-digit number, with Sunday as 0	0
%x	The preferred date representation for the current locale	02/28/02
%X	The preferred time representation for the current locale	13:04:24
%y	The year as a two-digit number	02
%Y	The year as a four-digit number	2002
%Z	The name of the current time zone	GMT Standard Time
%%	A % character in the output	%

SSI Variables

Within SSI parameter strings, the SSI parser performs shell-like variable expansions. This allows the contents of environment variables, including those set by the `set` SSI command, to be passed as part or all of the parameter values given to an SSI directive.

The inclusion is triggered by the use of a $ sign followed by the name of the variable you want to include. For example, you might have a different welcome line on your Web page for each day of the week. You could achieve this by storing the welcome lines in files called `Monday.txt`, `Tuesday.txt`, and so on and using the following lines in your SSI file:

```
<!--#config timefmt="%A" -->
<!--#include virtual="$DATE_LOCAL.txt" -->
```

Standard Include Variables

In addition to the variables in the standard environment, mod_include provides some additional variables that may be used by server-parsed documents (see Table E-4).

Table E-4. SSI Variables

Name	Description
DATE_GMT	The current date in Greenwich Mean Time.
DATE_LOCAL	The current date in the local time zone.
DOCUMENT_NAME	The filename (excluding directories) of the document requested by the user.
DOCUMENT_PATH_INFO	The additional path information that was passed to a document. See PATH_INFO previously and the AcceptPathInfo directive.
DOCUMENT_URI	The %-decoded URI (URL) of the document that corresponds to the URI originally requested by the user.
QUERY_STRING_UNESCAPED	The %-decoded query string sent by the client. See QUERY_STRING (which contains the encoded query string).
LAST_MODIFIED	The last modification date of the document requested by the user.
USER_NAME	The name of the user under that Apache is running. Determined by the User directive for the server when running as root but overridden for virtual hosts by the AssignUserID and ChildPerUserID directives when using the perchild MPM.

Condition SSI Evaluation

The Extended Server Side Include (XSSI) specification extends SSIs to allow the use of the conditional if, elif, else, and endif SSI commands. These may be used to control the flow of the parser.

The if and elif commands take an expr parameter, and the lines in the file after them (up to the next else, elif, or endif command) are only included if the expr parameter evaluates to true. This will be the case if the following happens:

- The expression is a nonempty string, such as expr="$REMOTE_USER".
- The expression is of the form expression1 comparison_operator expression2 where the comparison_operator is either =, !=, >, <, <=, >=, ||, or &&, and the relevant comparison gives the result as true. (!= means is not equal to, || signifies a Boolean OR, and && signifies a Boolean AND.) For example: expr="$REMOTE_USER = admin".

Variables may be used from any source, including environment variables set by PassEnv, SetEnv, SetEnvIf, BrowserMatch, and RewriteRule. The results of a previous regular expression match can also be retrieved from the variables $1 to $9.

The basic flow control elements are as follows:

```
<!--#if expr="test_condition" -->
<!--#elif expr="test_condition" -->
<!--#else -->
<!--#endif -->
```

The `if` element works like an `if` statement in a programming language. The `test_condition` is evaluated, and if the result is `true`, then the text until the next `elif`, `else`, or `endif` element is included in the output stream.

The `elif` or `else` statements are used to put the text into the output stream if the original `test_condition` was `false`. These elements are optional. The `endif` element ends the `if` element and is required.