Apache Module mod_alias

Description:	Provides for mapping different parts of the host filesystem in the document tree and for URL redirection
Status:	Base
Module Identifier:	alias_module
Source File:	mod_alias.c

Summary

The directives contained in this module allow for manipulation and control of URLs as requests arrive at the server. The `Alias` and `ScriptAlias` directives are used to map between URLs and filesystem paths. This allows for content which is not directly under the `DocumentRoot` served as part of the web document tree. The `ScriptAlias` directive has the additional effect of marking the target directory as containing only CGI scripts.

The `Redirect` directives are used to instruct clients to make a new request with a different URL. They are often used when a resource has moved to a new location.

Topics

Directives

See also

- `mod_rewrite`
- Mapping URLs to the filesystem[1]

Alias Directive

Description:	Maps URLs to filesystem locations
Syntax:	`Alias` *`URL-path file-path|directory-path`*
Context:	server config, virtual host
Status:	Base
Module:	mod_alias

The `Alias` directive allows documents to be stored in the local filesystem other than under the `DocumentRoot`. URLs with a (%-decoded) path beginning with *url-path* will be mapped to local files beginning with *directory-path*.

Example:

```
Alias /image /ftp/pub/image
```

A request for http://myserver/image/foo.gif would cause the server to return the file /ftp/pub/image/foo.gif.

Note that if you include a trailing / on the *url-path* then the server will require a trailing / in order to expand the alias. That is, if you use `Alias /icons/ /usr/local/apache/icons/` then the url `/icons` will not be aliased.

Note that you may need to specify additional `<Directory>` sections which cover the *destination* of aliases. Aliasing occurs before `<Directory>` sections are checked, so only the destination of aliases are affected. (Note however `<Location>` sections are run through once before aliases are performed, so they will apply.)

AliasMatch Directive

Description:	Maps URLs to filesystem locations using regular expressions
Syntax:	`AliasMatch` *`regex file-path`*`|`*`directory-path`*
Context:	server config, virtual host
Status:	Base
Module:	mod_alias

This directive is equivalent to `Alias`, but makes use of standard regular expressions, instead of simple prefix matching. The supplied regular expression is matched against the URL-path, and if it matches, the server will substitute any parenthesized matches into the given string and use it as a filename. For example, to activate the `/icons` directory, one might use:

```
AliasMatch ^/icons(.*) /usr/local/apache/icons$1
```

Redirect Directive

Description:	Sends an external redirect asking the client to fetch a different URL
Syntax:	`Redirect [`*`status`*`]` *`URL-path URL`*
Context:	server config, virtual host, directory, .htaccess
Override:	FileInfo
Status:	Base
Module:	mod_alias

The Redirect directive maps an old URL into a new one. The new URL is returned to the client which attempts to fetch it again with the new address. *URL-path* a (%-decoded) path; any requests for documents beginning with this path will be returned a redirect error to a new (%-encoded) URL beginning with *URL*.

Example:

```
Redirect /service http://foo2.bar.com/service
```

If the client requests http://myserver/service/foo.txt, it will be told to access http://foo2.bar.com/service/foo.txt instead.

Note

Redirect directives take precedence over Alias and ScriptAlias directives, irrespective of their ordering in the configuration file. Also, *URL-path* must be an absolute path, not a relative path, even when used with .htaccess files or inside of `<Directory>` sections.

If no *status* argument is given, the redirect will be "temporary" (HTTP status 302). This indicates to the client that the resource has moved temporarily. The *status* argument can be used to return other HTTP status codes:

permanent
Returns a permanent redirect status (301) indicating that the resource has moved permanently.

temp
Returns a temporary redirect status (302). This is the default.

seeother
Returns a "See Other" status (303) indicating that the resource has been replaced.

gone
Returns a "Gone" status (410) indicating that the resource has been permanently removed. When this status is used the *URL* argument should be omitted.

Other status codes can be returned by giving the numeric status code as the value of *status*. If the status is between 300 and 399, the *URL* argument must be present, otherwise it must be omitted. Note that the status must be known to the Apache code (see the function `send_error_response` in http_protocol.c).

Example:

```
Redirect permanent /one http://example.com/two
Redirect 303 /three http://example.com/other
```

RedirectMatch Directive

Description:	Sends an external redirect based on a regular expression match of the current URL
Syntax:	`RedirectMatch [`*`status`*`]` *`regex URL`*
Context:	server config, virtual host, directory, .htaccess
Override:	FileInfo
Status:	Base
Module:	mod_alias

This directive is equivalent to `Redirect`, but makes use of standard regular expressions, instead of simple prefix matching. The supplied regular expression is matched against the URL-path, and if it matches, the server will substitute any parenthesized matches into the given string and use it as a filename. For example, to redirect all GIF files to like-named JPEG files on another server, one might use:

```
RedirectMatch (.*)\.gif$ http://www.anotherserver.com$1.jpg
```

RedirectPermanent Directive

Description:	Sends an external permanent redirect asking the client to fetch a different URL
Syntax:	`RedirectPermanent` *`URL-path URL`*
Context:	server config, virtual host, directory, .htaccess
Override:	FileInfo
Status:	Base

Module:	mod_alias

This directive makes the client know that the Redirect is permanent (status 301). Exactly equivalent to `Redirect permanent`.

RedirectTemp Directive

Description:	Sends an external temporary redirect asking the client to fetch a different URL
Syntax:	`RedirectTemp` *`URL-path URL`*
Context:	server config, virtual host, directory, .htaccess
Override:	FileInfo
Status:	Base
Module:	mod_alias

This directive makes the client know that the Redirect is only temporary (status 302). Exactly equivalent to `Redirect temp`.

ScriptAlias Directive

Description:	Maps a URL to a filesystem location and designates the target as a CGI script
Syntax:	`ScriptAlias` *`URL-path file-path`*`|`*`directory-path`*
Context:	server config, virtual host
Status:	Base
Module:	mod_alias

The `ScriptAlias` directive has the same behavior as the `Alias` directive, except that in addition it marks the target directory as containing CGI scripts that will be processed by `mod_cgi`'s cgi-script handler. URLs with a (%-decoded) path beginning with *URL-path* will be mapped to scripts beginning with the second argument which is a full pathname in the local filesystem.

Example:

```
ScriptAlias /cgi-bin/ /web/cgi-bin/
```

A request for `http://myserver/cgi-bin/foo` would cause the server to run the script `/web/cgi-bin/foo`.

ScriptAliasMatch Directive

Description:	Maps a URL to a filesystem location using a regular expression and designates the target as a CGI script
Syntax:	`ScriptAliasMatch` *`regex file-path`*`|`*`directory-path`*
Context:	server config, virtual host
Status:	Base
Module:	mod_alias

This directive is equivalent to `ScriptAlias`, but makes use of standard regular expressions, instead of simple prefix matching. The supplied regular expression is matched against the URL-path, and if it matches, the server will substitute any parenthesized matches into the given string and use it as a filename. For example, to activate the standard `/cgi-bin`, one might use:

```
ScriptAliasMatch ^/cgi-bin(.*) /usr/local/apache/cgi-bin$1
```

URI References

[1] http://httpd.apache.org/docs-2.1/urlmapping.html