

1 Apache::Server - Perl API for for Apache server record accessors

1.1 Synopsis

```
use Apache::Server ();
```

META: to be completed

1.2 Description

META: to be completed

1.3 API

`Apache::Server` provides the following functions and/or methods:

1.3.1 process

META: Autogenerated - needs to be reviewed/completed

The process this server is running in

```
$proc = $s->process();
```

- **arg1: \$s (Apache::Server)**
- **ret: \$proc (Apache::Process)**

1.3.2 next

META: Autogenerated - needs to be reviewed/completed

The next server in the list (if there are vhosts)

```
$next_s = $s->next();
```

- **arg1: \$s (Apache::Server)**
- **ret: \$next_s (Apache::Server)**

For example the following code traverses all the servers, starting from the base server and continuing to vhost servers, counting all vhosts:

```
use Apache::Server ();
use Apache::ServerUtil ();
my $server = Apache->server;
my $vhosts = 0;
for (my $s = $server->next; $s; $s = $s->next) {
    $vhosts++;
}
```

1.3.3 *server_admin*

Get/set the server admin value

```
$server_admin = $s->server_admin();  
$prev_server_admin = $s->server_admin($new_server_admin);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_server_admin (string)**

If passed, sets the new server_admin.

- **ret: \$server_admin (string)**

Returns the server_admin setting.

If \$new_server_admin is passed returns the setting before the change.

1.3.4 *server_hostname*

Get/set the server hostname value

```
$server_hostname = $s->server_hostname();  
$prev_server_hostname = $s->server_hostname($new_server_hostname);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_server_hostname (string)**

If passed, sets the new server_hostname.

- **ret: \$server_hostname (string)**

Returns the server_hostname setting.

If \$new_server_hostname is passed returns the setting before the change.

1.3.5 *port*

META: Autogenerated - needs to be reviewed/completed

Get/set the port value

```
$port = $s->port();  
$prev_port = $s->port($new_port);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_port (string)**

1.3.6 error_fname

If passed, sets the new port.

- **ret: \$port (string)**

Returns the port setting.

If \$new_port is passed returns the setting before the change.

1.3.6 error_fname

META: Autogenerated - needs to be reviewed/completed

Get/set the error_fname value

```
$error_fname = $s->error_fname();  
$prev_error_fname = $s->error_fname($new_error_fname);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_error_fname (string)**

If passed, sets the new error_fname.

- **ret: \$error_fname (string)**

Returns the error_fname setting.

If \$new_error_fname is passed returns the setting before the change.

1.3.7 loglevel

META: Autogenerated - needs to be reviewed/completed

Get/set the log level value

```
$loglevel = $s->loglevel();  
$prev_loglevel = $s->loglevel($new_loglevel);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_loglevel (string)**

If passed, sets the new loglevel.

- **ret: \$loglevel (string)**

Returns the loglevel setting.

If \$new_loglevel is passed returns the setting before the change.

1.3.8 *is_virtual*

META: Autogenerated - needs to be reviewed/completed

Get/set the is_virtual value

```
$is_virtual = $s->is_virtual();
$prev_is_virtual = $s->is_virtual($new_is_virtual);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_is_virtual (string)**

If passed, sets the new is_virtual.

META: this is wrong, it should be a read only accessor

- **ret: \$is_virtual (string)**

Returns the is_virtual setting.

If \$new_is_virtual is passed returns the setting before the change.

1.3.9 *module_config*

META: Autogenerated - needs to be reviewed/completed

Get/set config vector containing pointers to modules' per-server config structures.

```
$module_config = $s->module_config();
$prev_module_config = $s->module_config($new_module_config);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: new_module_config (Apache::ConfVector)**

If passed, sets the new module_config.

- **ret: \$module_config (Apache::ConfVector)**

Returns the module_config setting.

If \$new_module_config is passed returns the setting before the change.

1.3.10 *lookup_defaults*

META: Autogenerated - needs to be reviewed/completed

Get/set the lookup_defaults value. MIME type info, etc., before we start checking per-directory info.

1.3.11 *addr*s

```
$lookup_defaults = $s->lookup_defaults();  
$prev_lookup_defaults = $s->lookup_defaults($new_lookup_defaults);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_lookup_defaults (Apache::ConfVector)**

If passed, sets the new lookup_defaults.

- **ret: \$lookup_defaults (Apache::ConfVector)**

Returns the lookup_defaults setting.

If \$new_lookup_defaults is passed returns the setting before the change.

1.3.11 *addr*s

META: Autogenerated - needs to be reviewed/completed

Get/set the addr's value

```
$addr = $s->addr();  
$prev_addr = $s->addr($new_addr);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_addr (Apache::ServerAddr)**

If passed, sets the new addr's.

- **ret: \$addr (Apache::ServerAddr)**

Returns the addr's setting.

If \$new_addr is passed returns the setting before the change.

1.3.12 *timeout*

META: Autogenerated - needs to be reviewed/completed

Get/set the timeout, as an apr interval, before we give up

```
$timeout = $s->timeout();  
$prev_timeout = $s->timeout($new_timeout);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_timeout (string)**

If passed, sets the new timeout.

- **ret: \$timeout (string)**

Returns the timeout setting.

If \$new_timeout is passed returns the setting before the change.

1.3.13 keep_alive_timeout

META: Autogenerated - needs to be reviewed/completed

Get/set the apr interval we will wait for another request

```
$keep_alive_timeout = $s->keep_alive_timeout();  
$prev_keep_alive_timeout = $s->keep_alive_timeout($new_keep_alive_timeout);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_keep_alive_timeout (string)**

If passed, sets the new keep_alive_timeout.

- **ret: \$keep_alive_timeout (string)**

Returns the keep_alive_timeout setting.

If \$new_keep_alive_timeout is passed returns the setting before the change.

1.3.14 keep_alive_max

META: Autogenerated - needs to be reviewed/completed

Get/set maximum requests per connection

```
$keep_alive_max = $s->keep_alive_max();  
$prev_keep_alive_max = $s->keep_alive_max($new_keep_alive_max);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_keep_alive_max (string)**

If passed, sets the new keep_alive_max.

- **ret: \$keep_alive_max (string)**

Returns the keep_alive_max setting.

If \$new_keep_alive_max is passed returns the setting before the change.

1.3.15 keep_alive

META: Autogenerated - needs to be reviewed/completed

Use persistent connections?

```
$keep_alive = $s->keep_alive();  
$prev_keep_alive = $s->keep_alive($new_keep_alive);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_keep_alive (string)**

If passed, sets the new keep_alive.

- **ret: \$keep_alive (string)**

Returns the keep_alive setting.

If \$new_keep_alive is passed returns the setting before the change.

1.3.16 path

META: Autogenerated - needs to be reviewed/completed

Get/set pathname for ServerPath

```
$path = $s->path();  
$prev_path = $s->path($new_path);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_path (string)**

If passed, sets the new path.

- **ret: \$path (string)**

Returns the path setting.

If \$new_path is passed returns the setting before the change.

1.3.17 names

META: Autogenerated - needs to be reviewed/completed

Get/set normal names for ServerAlias servers

```
$names = $s->names();  
$prev_names = $s->names($new_names);
```


- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_names (APR::ArrayHeader)**

If passed, sets the new names.

- **ret: \$names (APR::ArrayHeader)**

Returns the names setting.

If \$new_names is passed returns the setting before the change.

1.3.18 wild_names

META: Autogenerated - needs to be reviewed/completed

Wildcarded names for ServerAlias servers

```
$wild_names = $s->wild_names();
$prev_wild_names = $s->wild_names($new_wild_names);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_wild_names (APR::ArrayHeader)**

If passed, sets the new wild_names.

- **ret: \$wild_names (APR::ArrayHeader)**

Returns the wild_names setting.

If \$new_wild_names is passed returns the setting before the change.

1.3.19 limit_req_line

META: Autogenerated - needs to be reviewed/completed

Get/set limit on size of the HTTP request line

```
$limit_req_line = $s->limit_req_line();
$prev_limit_req_line = $s->limit_req_line($new_limit_req_line);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_limit_req_line (string)**

If passed, sets the new limit_req_line.

- **ret: \$limit_req_line (string)**

Returns the limit_req_line setting.

If `$new_limit_req_line` is passed returns the setting before the change.

1.3.20 limit_req_fieldsize

META: Autogenerated - needs to be reviewed/completed

limit on size of any request header field

```
$limit_req_fieldsize = $s->limit_req_fieldsize();  
$prev_limit_req_fieldsize = $s->limit_req_fieldsize($new_limit_req_fieldsize);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_limit_req_fieldsize (string)**

If passed, sets the new `limit_req_fieldsize`.

- **ret: \$limit_req_fieldsize (string)**

Returns the `limit_req_fieldsize` setting.

If `$new_limit_req_fieldsize` is passed returns the setting before the change.

1.3.21 limit_req_fields

META: Autogenerated - needs to be reviewed/completed

Get/set limit on number of request header fields

```
$limit_req_fields = $s->limit_req_fields();  
$prev_limit_req_fields = $s->limit_req_fields($new_limit_req_fields);
```

- **arg1: \$s (Apache::Server)**
- **opt arg2: \$new_limit_req_fields (string)**

If passed, sets the new `limit_req_fields`.

- **ret: \$limit_req_fields (string)**

Returns the `limit_req_fields` setting.

If `$new_limit_req_fields` is passed returns the setting before the change.

1.4 See Also

mod_perl 2.0 documentation.

1.5 Copyright

mod_perl 2.0 and its core modules are copyrighted under The Apache Software License, Version 1.1.

1.6 Authors

The mod_perl development team and numerous contributors.

Table of Contents:

1	Apache::Server - Perl API for for Apache server record accessors	1
1.1	Synopsis	2
1.2	Description	2
1.3	API	2
1.3.1	process	2
1.3.2	next	2
1.3.3	server_admin	3
1.3.4	server_hostname	3
1.3.5	port	3
1.3.6	error_fname	4
1.3.7	loglevel	4
1.3.8	is_virtual	5
1.3.9	module_config	5
1.3.10	lookup_defaults	5
1.3.11	addrs	6
1.3.12	timeout	6
1.3.13	keep_alive_timeout	7
1.3.14	keep_alive_max	7
1.3.15	keep_alive	8
1.3.16	path	8
1.3.17	names	8
1.3.18	wild_names	9
1.3.19	limit_req_line	9
1.3.20	limit_req_fieldsize	10
1.3.21	limit_req_fields	10
1.4	See Also	10
1.5	Copyright	11
1.6	Authors	11