

NAME

HTTP::Config – Configuration for request and response objects

VERSION

version 6.22

SYNOPSIS

```
use HTTP::Config;
my $c = HTTP::Config->new;
$c->add(m_domain => ".example.com", m_scheme => "http", verbose => 1);

use HTTP::Request;
my $request = HTTP::Request->new(GET => "http://www.example.com");

if (my @m = $c->matching($request)) {
    print "Yadayada\n" if $m[0]->{verbose};
}
```

DESCRIPTION

An `HTTP::Config` object is a list of entries that can be matched against request or request/response pairs. Its purpose is to hold configuration data that can be looked up given a request or response object.

Each configuration entry is a hash. Some keys specify matching to occur against attributes of request/response objects. Other keys can be used to hold user data.

The following methods are provided:

`$conf = HTTP::Config->new`

Constructs a new empty `HTTP::Config` object and returns it.

`$conf->entries`

Returns the list of entries in the configuration object. In scalar context returns the number of entries.

`$conf->empty`

Return true if there are no entries in the configuration object. This is just a shorthand for `not $conf->entries`.

`$conf->add(%matchspec, %other)`

`$conf->add(\%entry)`

Adds a new entry to the configuration. You can either pass separate key/value pairs or a hash reference.

`$conf->remove(%spec)`

Removes (and returns) the entries that have matches for all the key/value pairs in `%spec`. If `%spec` is empty this will match all entries; so it will empty the configuration object.

`$conf->matching($uri, $request, $response)`

`$conf->matching($uri)`

`$conf->matching($request)`

`$conf->matching($response)`

Returns the entries that match the given `$uri`, `$request` and `$response` triplet.

If called with a single `$request` object then the `$uri` is obtained by calling its `'uri_canonical'` method. If called with a single `$response` object, then the request object is obtained by calling its `'request'` method; and then the `$uri` is obtained as if a single `$request` was provided.

The entries are returned with the most specific matches first. In scalar context returns the most specific match or `undef` in none match.

`$conf->add_item($item, %matchspec)`

`$conf->remove_items(%spec)`

```
$conf->matching_items( $uri, $request, $response )
```

Wrappers that hides the entries themselves.

Matching

The following keys on a configuration entry specify matching. For all of these you can provide an array of values instead of a single value. The entry matches if at least one of the values in the array matches.

Entries that require match against a response object attribute will never match unless a response object was provided.

```
m_scheme => $scheme
```

Matches if the URI uses the specified scheme; e.g. “http”.

```
m_secure => $bool
```

If \$bool is TRUE; matches if the URI uses a secure scheme. If \$bool is FALSE; matches if the URI does not use a secure scheme. An example of a secure scheme is “https”.

```
m_host_port => “$hostname:$port”
```

Matches if the URI’s host_port method return the specified value.

```
m_host => $hostname
```

Matches if the URI’s host method returns the specified value.

```
m_port => $port
```

Matches if the URI’s port method returns the specified value.

```
m_domain => “.$domain”
```

Matches if the URI’s host method return a value that within the given domain. The hostname “www.example.com” will for instance match the domain “.com”.

```
m_path => $path
```

Matches if the URI’s path method returns the specified value.

```
m_path_prefix => $path
```

Matches if the URI’s path is the specified path or has the specified path as prefix.

```
m_path_match => $Regexp
```

Matches if the regular expression matches the URI’s path. Eg. qr/\.html\$/.

```
m_method => $method
```

Matches if the request method matches the specified value. Eg. “GET” or “POST”.

```
m_code => $digit
```

```
m_code => $status_code
```

Matches if the response status code matches. If a single digit is specified; matches for all response status codes beginning with that digit.

```
m_proxy => $url
```

Matches if the request is to be sent to the given Proxy server.

```
m_media_type => “*/*”
```

```
m_media_type => “text/*”
```

```
m_media_type => “html”
```

```
m_media_type => “xhtml”
```

```
m_media_type => “text/html”
```

Matches if the response media type matches.

With a value of “html” matches if \$response->content_is_html returns TRUE. With a value of “xhtml” matches if \$response->content_is_xhtml returns TRUE.

```
m_uri_-$method => undef
```

Matches if the URI object provides the method.

`m_uri__$method=> $string`

Matches if the URI's `$method` method returns the given value.

`m_header__$field=> $string`

Matches if either the request or the response have a header `$field` with the given value.

`m_response_attr__$key=> undef`

`m_response_attr__$key=> $string`

Matches if the response object has that key, or the entry has the given value.

SEE ALSO

URI, HTTP::Request, HTTP::Response

AUTHOR

Gisle Aas <gisle@activestate.com>

COPYRIGHT AND LICENSE

This software is copyright (c) 1994–2017 by Gisle Aas.

This is free software; you can redistribute it and/or modify it under the same terms as the Perl 5 programming language system itself.