NAME

HTTP::Status - HTTP Status code processing

SYNOPSIS

```
use HTTP::Status qw(:constants :is status_message);

if ($rc != HTTP_OK) {
print status_message($rc), "\n";
}

if (is_success($rc)) { ... }
if (is_error($rc)) { ... }
if (is_redirect($rc)) { ... }
```

DESCRIPTION

HTTP::Status is a library of routines for defining and classifying HTTP status codes for libwww-perl. Status codes are used to encode the overall outcome of an HTTP response message. Codes correspond to those defined in RFC 2616 and RFC 2518.

CONSTANTS

The following constant functions can be used as mnemonic status code names. None of these are exported by default. Use the `:constants` tag to import them all.

```
HTTP_CONTINUE (100)
HTTP_SWITCHING_PROTOCOLS (101)
HTTP_PROCESSING (102)

HTTP_OK (200)
HTTP_CREATED (201)
HTTP_ACCEPTED (202)
HTTP_NON_AUTHORITATIVE_INFORMATION (203)
HTTP_NO_CONTENT (204)
HTTP_RESET_CONTENT (205)
HTTP_PARTIAL_CONTENT (206)
HTTP_MULTI_STATUS (207)
HTTP_ALREADY_REPORTED (208)

HTTP_MULTIPLE_CHOICES (300)
HTTP_MOVED_PERMANENTLY (301)
HTTP_FOUND (302)
HTTP_SEE_OTHER (303)
HTTP_NOT_MODIFIED (304)
HTTP_USE_PROXY (305)
HTTP_TEMPORARY_REDIRECT (307)

HTTP_BAD_REQUEST (400)
HTTP_UNAUTHORIZED (401)
HTTP_PAYMENT_REQUIRED (402)
HTTP_FORBIDDEN (403)
HTTP_NOT_FOUND (404)
HTTP_METHOD_NOT_ALLOWED (405)
HTTP_NOT_ACCEPTABLE (406)
HTTP_PROXY_AUTHENTICATION_REQUIRED (407)
HTTP_REQUEST_TIMEOUT (408)
HTTP_CONFLICT (409)
HTTP_GONE (410)
HTTP_LENGTH_REQUIRED (411)
```

```
HTTP_PRECONDITION_FAILED (412)
HTTP_REQUEST_ENTITY_TOO_LARGE (413)
HTTP_REQUEST_URI_TOO_LARGE (414)
HTTP_UNSUPPORTED_MEDIA_TYPE (415)
HTTP_REQUEST_RANGE_NOT_SATISFIABLE (416)
HTTP_EXPECTATION_FAILED (417)
HTTP_I_AM_A_TEAPOT (418)
HTTP_UNPROCESSABLE_ENTITY (422)
HTTP_LOCKED (423)
HTTP_FAILED_DEPENDENCY (424)
HTTP_NO_CODE (425)
HTTP_UPGRADE_REQUIRED (426)
HTTP_PRECONDITION_REQUIRED (428)
HTTP_TOO_MANY_REQUESTS (429)
HTTP_REQUEST_HEADER_FIELDS_TOO_LARGE (431)
HTTP_RETRY_WITH (449)

HTTP_INTERNAL_SERVER_ERROR (500)
HTTP_NOT_IMPLEMENTED (501)
HTTP_BAD_GATEWAY (502)
HTTP_SERVICE_UNAVAILABLE (503)
HTTP_GATEWAY_TIMEOUT (504)
HTTP_HTTP_VERSION_NOT_SUPPORTED (505)
HTTP_VARIANT_ALSO_NEGOTIATES (506)
HTTP_INSUFFICIENT_STORAGE (507)
HTTP_BANDWIDTH_LIMIT_EXCEEDED (509)
HTTP_NOT_EXTENDED (510)
HTTP_NETWORK_AUTHENTICATION_REQUIRED (511)
```

FUNCTIONS

The following additional functions are provided. Most of them are exported by default. The `:is` import tag can be used to import all the classification functions.

status_message(`$code`)

The *status_message()* function will translate status codes to human readable strings. The string is the same as found in the constant names above. If the `$code` is unknown, then `undef` is returned.

is_info(`$code`)

Return TRUE if `$code` is an *Informational* status code (1xx). This class of status code indicates a provisional response which can't have any content.

is_success(`$code`)

Return TRUE if `$code` is a *Successful* status code (2xx).

is_redirect(`$code`)

Return TRUE if `$code` is a *Redirection* status code (3xx). This class of status code indicates that further action needs to be taken by the user agent in order to fulfill the request.

is_error(`$code`)

Return TRUE if `$code` is an *Error* status code (4xx or 5xx). The function returns TRUE for both client and server error status codes.

is_client_error(`$code`)

Return TRUE if `$code` is a *Client Error* status code (4xx). This class of status code is intended for cases in which the client seems to have erred.

This function is **not** exported by default.

is_server_error(`$code`)

Return TRUE if `$code` is a *Server Error* status code (5xx). This class of status codes is intended for cases in which the server is aware that it has erred or is incapable of performing the request.

This function is **not** exported by default.

BUGS

For legacy reasons all the `HTTP_` constants are exported by default with the prefix `RC_`. It's recommended to use explicit imports and the `:constants` tag instead of relying on this.