NAME

HTTP::Negotiate – choose a variant to serve

SYNOPSIS

```
 use HTTP::Negotiate qw(choose);

 #  ID       QS     Content-Type   Encoding Char-Set        Lang   Size
 $variants =
  [['var1',  1.000, 'text/html',   undef,   'iso-8859-1',   'en',   3000],
   ['var2',  0.950, 'text/plain',  'gzip',  'us-ascii',     'no',    400],
   ['var3',  0.3,   'image/gif',   undef,   undef,          undef, 43555],
  ];

 @preferred = choose($variants, $request_headers);
 $the_one   = choose($variants);
```

DESCRIPTION

This module provides a complete implementation of the HTTP content negotiation algorithm specified in *draft−ietf−http−v11−spec−00.ps* chapter 12. Content negotiation allows for the selection of a preferred content representation based upon attributes of the negotiable variants and the value of the various Accept* header fields in the request.

The variants are ordered by preference by calling the function *choose()*.

The first parameter is reference to an array of the variants to choose among. Each element in this array is an array with the values [$id, `$qs`, `$content_type`, `$content_encoding`, `$charset`, `$content_language`, `$content_length`] whose meanings are described below. The `$content_encoding` and `$content_language` can be either a single scalar value or an array reference if there are several values.

The second optional parameter is either a HTTP::Headers or a HTTP::Request object which is searched for "Accept*" headers. If this parameter is missing, then the accept specification is initialized from the CGI environment variables HTTP_ACCEPT, HTTP_ACCEPT_CHARSET, HTTP_ACCEPT_ENCODING and HTTP_ACCEPT_LANGUAGE.

In an array context, *choose()* returns a list of [variant identifier, calculated quality, size] tuples. The values are sorted by quality, highest quality first. If the calculated quality is the same for two variants, then they are sorted by size (smallest first). *E.g.*:

```
  (['var1', 1, 2000], ['var2', 0.3, 512], ['var3', 0.3, 1024]);
```

Note that also zero quality variants are included in the return list even if these should never be served to the client.

In a scalar context, it returns the identifier of the variant with the highest score or `undef` if none have non-zero quality.

If the `$HTTP::Negotiate::DEBUG` variable is set to TRUE, then a lot of noise is generated on STDOUT during evaluation of *choose()*.

VARIANTS

A variant is described by a list of the following values. If the attribute does not make sense or is unknown for a variant, then use `undef` instead.

identifier
: This is a string that you use as the name for the variant. This identifier for the preferred variants returned by *choose()*.

qs
: This is a number between 0.000 and 1.000 that describes the "source quality". This is what *draft−ietf−http−v11−spec−00.ps* says about this value:

  Source quality is measured by the content provider as representing the amount of degradation from the original source. For example, a picture in JPEG form would have a lower qs when translated to the

XBM format, and much lower qs when translated to an ASCII-art representation. Note, however, that this is a function of the source – an original piece of ASCII-art may degrade in quality if it is captured in JPEG form. The qs values should be assigned to each variant by the content provider; if no qs value has been assigned, the default is generally "qs=1".

content-type
This is the media type of the variant. The media type does not include a charset attribute, but might contain other parameters. Examples are:

```
text/html
text/html;version=2.0
text/plain
image/gif
image/jpg
```

content-encoding
This is one or more content encodings that has been applied to the variant. The content encoding is generally used as a modifier to the content media type. The most common content encodings are:

```
gzip
compress
```

content-charset
This is the character set used when the variant contains text. The charset value should generally be `undef` or one of these:

```
us-ascii
iso-8859-1 ... iso-8859-9
iso-2022-jp
iso-2022-jp-2
iso-2022-kr
unicode-1-1
unicode-1-1-utf-7
unicode-1-1-utf-8
```

content-language
This describes one or more languages that are used in the variant. Language is described like this in *draft−ietf−http−v11−spec−00.ps*: A language is in this context a natural language spoken, written, or otherwise conveyed by human beings for communication of information to other human beings. Computer languages are explicitly excluded.

The language tags are defined by RFC 3066. Examples are:

```
no               Norwegian
en               International English
en-US            US English
en-cockney
```

content-length
This is the number of bytes used to represent the content.

ACCEPT HEADERS

The following Accept* headers can be used for describing content preferences in a request (This description is an edited extract from *draft−ietf−http−v11−spec−00.ps*):

Accept
This header can be used to indicate a list of media ranges which are acceptable as a response to the request. The "*" character is used to group media types into ranges, with "*/*" indicating all media types and "type/*" indicating all subtypes of that type.

The parameter q is used to indicate the quality factor, which represents the user's preference for that range of media types. The parameter mbx gives the maximum acceptable size of the response content.

The default values are: q=1 and mbx=infinity. If no Accept header is present, then the client accepts all media types with q=1.

For example:

```
Accept: audio/*;q=0.2;mbx=200000, audio/basic
```

would mean: "I prefer audio/basic (of any size), but send me any audio type if it is the best available after an 80% mark-down in quality and its size is less than 200000 bytes"

Accept-Charset
Used to indicate what character sets are acceptable for the response. The "us-ascii" character set is assumed to be acceptable for all user agents. If no Accept-Charset field is given, the default is that any charset is acceptable. Example:

```
Accept-Charset: iso-8859-1, unicode-1-1
```

Accept-Encoding
Restricts the Content-Encoding values which are acceptable in the response. If no Accept-Encoding field is present, the server may assume that the client will accept any content encoding. An empty Accept-Encoding means that no content encoding is acceptable. Example:

```
Accept-Encoding: compress, gzip
```

Accept-Language
This field is similar to Accept, but restricts the set of natural languages that are preferred in a response. Each language may be given an associated quality value which represents an estimate of the user's comprehension of that language. For example:

```
Accept-Language: no, en-gb;q=0.8, de;q=0.55
```

would mean: "I prefer Norwegian, but will accept British English (with 80% comprehension) or German (with 55% comprehension).

COPYRIGHT

AUTHOR

Gisle Aas <gisle@aas.no>