

**NAME**

HTML::HeadParser – Parse <HEAD> section of a HTML document

**SYNOPSIS**

```
require HTML::HeadParser;
$p = HTML::HeadParser->new;
$p->parse($text) and print "not finished";

$p->header('Title')           # to access <title>....</title>
$p->header('Content-Base')    # to access <base href="http://...">
$p->header('Foo')             # to access <meta http-equiv="Foo" content="...">
$p->header('X-Meta-Author')   # to access <meta name="author" content="...">
$p->header('X-Meta-Charset')  # to access <meta charset="...">
```

**DESCRIPTION**

The `HTML::HeadParser` is a specialized (and lightweight) `HTML::Parser` that will only parse the <HEAD>...</HEAD> section of an HTML document. The **parse()** method will return a FALSE value as soon as some <BODY> element or body text are found, and should not be called again after this.

Note that the `HTML::HeadParser` might get confused if raw undecoded UTF-8 is passed to the **parse()** method. Make sure the strings are properly decoded before passing them on.

The `HTML::HeadParser` keeps a reference to a header object, and the parser will update this header object as the various elements of the <HEAD> section of the HTML document are recognized. The following header fields are affected:

**Content-Base:**

The *Content-Base* header is initialized from the <base href="..."> element.

**Title:**

The *Title* header is initialized from the <title>...</title> element.

**Isindex:**

The *Isindex* header will be added if there is a <isindex> element in the <head>. The header value is initialized from the *prompt* attribute if it is present. If no *prompt* attribute is given it will have '?' as the value.

**X-Meta-Foo:**

All <meta> elements containing a name attribute will result in headers using the prefix X-Meta- appended with the value of the name attribute as the name of the header, and the value of the content attribute as the pushed header value.

<meta> elements containing a http-equiv attribute will result in headers as in above, but without the X-Meta- prefix in the header name.

<meta> elements containing a charset attribute will result in an X-Meta-Charset header, using the value of the charset attribute as the pushed header value.

The ':' character can't be represented in header field names, so if the meta element contains this char it's substituted with '-' before forming the field name.

**METHODS**

The following methods (in addition to those provided by the superclass) are available:

```
$hp = HTML::HeadParser->new
```

```
$hp = HTML::HeadParser->new( $header )
```

The object constructor. The optional *\$header* argument should be a reference to an object that implement the **header()** and **push\_header()** methods as defined by the `HTTP::Headers` class. Normally it will be of some class that is a or delegates to the `HTTP::Headers` class.

If no *\$header* is given `HTML::HeadParser` will create an `HTTP::Headers` object by itself (initially empty).

```
$hp->header;
```

Returns a reference to the header object.

```
$hp->header( $key )
```

Returns a header value. It is just a shorter way to write `$hp->header->header( $key )`.

### EXAMPLE

```
$h = HTTP::Headers->new;
$p = HTML::HeadParser->new($h);
$p->parse(<<EOT);
<title>Stupid example</title>
<base href="http://www.linpro.no/lwp/">
Normal text starts here.
EOT
undef $p;
print $h->title;    # should print "Stupid example"
```

### SEE ALSO

HTML::Parser, HTTP::Headers

The HTTP::Headers class is distributed as part of the *libwww-perl* package. If you don't have that distribution installed you need to provide the `$header` argument to the HTML::HeadParser constructor with your own object that implements the documented protocol.

### COPYRIGHT

Copyright 1996–2001 Gisle Aas. All rights reserved.

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