### **NAME**

WWW::RobotRules - database of robots.txt-derived permissions

#### **SYNOPSIS**

```
use WWW::RobotRules;
my $rules = WWW::RobotRules->new('MOMspider/1.0');
use LWP::Simple qw(get);
{
    my $url = "http://some.place/robots.txt";
    my $robots_txt = get $url;
    $rules->parse($url, $robots_txt) if defined $robots_txt;
}
{
    my $url = "http://some.other.place/robots.txt";
    my $robots_txt = get $url;
    $rules->parse($url, $robots_txt) if defined $robots_txt;
}
# Now we can check if a URL is valid for those servers
# whose "robots.txt" files we've gotten and parsed:
if($rules->allowed($url)) {
    $c = get $url;
    ...
}
```

### DESCRIPTION

This module parses /robots.txt files as specified in "A Standard for Robot Exclusion", at <a href="http://www.robotstxt.org/wc/norobots.html">http://www.robotstxt.org/wc/norobots.html</a> Webmasters can use the /robots.txt file to forbid conforming robots from accessing parts of their web site.

The parsed files are kept in a WWW::RobotRules object, and this object provides methods to check if access to a given URL is prohibited. The same WWW::RobotRules object can be used for one or more parsed */robots.txt* files on any number of hosts.

The following methods are provided:

```
$rules = WWW::RobotRules->new($robot_name)
```

This is the constructor for WWW::RobotRules objects. The first argument given to *new()* is the name of the robot.

```
$rules->parse($robot_txt_url, $content, $fresh_until)
```

The *parse()* method takes as arguments the URL that was used to retrieve the */robots.txt* file, and the contents of the file.

```
$rules->allowed($uri)
```

Returns TRUE if this robot is allowed to retrieve this URL.

```
$rules->agent([$name])
```

Get/set the agent name. NOTE: Changing the agent name will clear the robots.txt rules and expire times out of the cache.

## ROBOTS.TXT

The format and semantics of the "/robots.txt" file are as follows (this is an edited abstract of <a href="http://www.robotstxt.org/wc/norobots.html">http://www.robotstxt.org/wc/norobots.html</a>):

The file consists of one or more records separated by one or more blank lines. Each record contains lines of the form

```
<field-name>: <value>
```

The field name is case insensitive. Text after the '#' character on a line is ignored during parsing. This is used for comments. The following <field-names> can be used:

### User-Agent

The value of this field is the name of the robot the record is describing access policy for. If more than one *User-Agent* field is present the record describes an identical access policy for more than one robot. At least one field needs to be present per record. If the value is '\*', the record describes the default access policy for any robot that has not not matched any of the other records.

The *User-Agent* fields must occur before the *Disallow* fields. If a record contains a *User-Agent* field after a *Disallow* field, that constitutes a malformed record. This parser will assume that a blank line should have been placed before that *User-Agent* field, and will break the record into two. All the fields before the *User-Agent* field will constitute a record, and the *User-Agent* field will be the first field in a new record.

### Disallow

The value of this field specifies a partial URL that is not to be visited. This can be a full path, or a partial path; any URL that starts with this value will not be retrieved

Unrecognized records are ignored.

#### ROBOTS.TXT EXAMPLES

The following example "/robots.txt" file specifies that no robots should visit any URL starting with "/cyberworld/map/" or "/tmp/":

```
User-agent: *
Disallow: /cyberworld/map/ # This is an infinite virtual URL space
Disallow: /tmp/ # these will soon disappear
```

This example "/robots.txt" file specifies that no robots should visit any URL starting with "/cyberworld/map/", except the robot called "cybermapper":

```
User-agent: *
Disallow: /cyberworld/map/ # This is an infinite virtual URL space
# Cybermapper knows where to go.
User-agent: cybermapper
Disallow:
```

This example indicates that no robots should visit this site further:

```
# go away
User-agent: *
Disallow: /
```

This is an example of a malformed robots.txt file.

```
# robots.txt for ancientcastle.example.com
# I've locked myself away.
User-agent: *
Disallow: /
# The castle is your home now, so you can go anywhere you like.
User-agent: Belle
Disallow: /west-wing/ # except the west wing!
# It's good to be the Prince...
User-agent: Beast
Disallow:
```

This file is missing the required blank lines between records. However, the intention is clear.

# WWW::RobotRules(3pm)

# **SEE ALSO**

 $LWP::RobotUA,\,WWW::RobotRules::AnyDBM\_File$ 

# **COPYRIGHT**

Copyright 1995-2009, Gisle Aas Copyright 1995, Martijn Koster

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