NAME

DBD::Gofer::Transport::Base - base class for DBD::Gofer client transports

SYNOPSIS

```
 my $remote_dsn = "..."
 DBI->connect("dbi:Gofer:transport=...;url=...;timeout=...;retry_limit=...;dsn=$remote_dsn",.
```

or, enable by setting the DBI_AUTOPROXY environment variable:

```
 export DBI_AUTOPROXY='dbi:Gofer:transport=...;url=...'
```

which will force *all* DBI connections to be made via that Gofer server.

DESCRIPTION

This is the base class for all DBD::Gofer client transports.

ATTRIBUTES

Gofer transport attributes can be specified either in the attributes parameter of the *connect()* method call, or in the DSN string. When used in the DSN string, attribute names don't have the `go_` prefix.

go_dsn

The full DBI DSN that the Gofer server should connect to on your behalf.

When used in the DSN it must be the last element in the DSN string.

go_timeout

A time limit for sending a request and receiving a response. Some drivers may implement sending and receiving as separate steps, in which case (currently) the timeout applies to each separately.

If a request needs to be resent then the timeout is restarted for each sending of a request and receiving of a response.

go_retry_limit

The maximum number of times an request may be retried. The default is 2.

go_retry_hook

This subroutine reference is called, if defined, for each response received where `$response`->err is true.

The subroutine is pass three parameters: the request object, the response object, and the transport object.

If it returns an undefined value then the default retry behaviour is used. See "RETRY ON ERROR" below.

If it returns a defined but false value then the request is not resent.

If it returns true value then the request is resent, so long as the number of retries does not exceed `go_retry_limit`.

RETRY ON ERROR

The default retry on error behaviour is:

```
 - Retry if the error was due to DBI_GOFER_RANDOM. See L<DBI::Gofer::Execute>.

 - Retry if $request->is_idempotent returns true. See L<DBI::Gofer::Request>.
```

A retry won't be allowed if the number of previous retries has reached `go_retry_limit`.

TRACING

Tracing of gofer requests and responses can be enabled by setting the `DBD_GOFER_TRACE` environment variable. A value of 1 gives a reasonably compact summary of each request and response. A value of 2 or more gives a detailed, and voluminous, dump.

The trace is written using DBI->*trace_msg()* and so is written to the default DBI trace output, which is usually STDERR.

METHODS

This section is currently far from complete.

response_retry_preference

```
 $retry = $transport->response_retry_preference($request, $response);
```

The response_retry_preference is called by DBD::Gofer when considering if a request should be retried after an error.

Returns true (would like to retry), false (must not retry), undef (no preference).

If a true value is returned in the form of a CODE ref then, if DBD::Gofer does decide to retry the request, it calls the code ref passing `$retry_count`, `$retry_limit`. Can be used for logging and/or to implement exponential backoff behaviour. Currently the called code must return using `return;` to allow for future extensions.

AUTHOR

Tim Bunce, <http://www.tim.bunce.name>

LICENCE AND COPYRIGHT

SEE ALSO

DBD::Gofer, DBI::Gofer::Request, DBI::Gofer::Response, DBI::Gofer::Execute.

and some example transports:

DBD::Gofer::Transport::stream

DBD::Gofer::Transport::http

DBI::Gofer::Transport::mod_perl