Richardson & Büren

ihr partner für kivitendo

Entwickler-Schulung 17.04.2015

kivitendo

Bernd Bleßmann

Thema

- Einfaches Debugging

Debugging – Wohin?

Einstellungen in 'config/kivitendo.conf'

```
[debug]
...
file_name = /tmp/kivitendo-debug.log
```

Debugging – Was?

```
[debug]
...
#global_level = NONE
global_level = ALL
```

Debugging – Was?

```
# Possible values include:
#   NONE   - no debug output (default)
#   INFO
#   DEBUG1
#   DEBUG2
#   QUERY              - Dump SQL queries (only in legacy code; see also "dbix_log4perl"
above)
#   TRACE              - Track function calls and returns
#   BACKTRACE_ON_ERROR - Print a function call backtrace when $form->error() is called
#   REQUEST_TIMER      - Log timing of HTTP requests
#   REQUEST            - Log each request. Careful! Passwords get filtered, but
#                        there may be confidential information being logged here
#   WARN               - warnings
#   ALL                - all possible debug messages
#
#   DEVEL              - sames as "INFO QUERY TRACE BACKTRACE_ON_ERROR REQUEST_TIMER"
#
# Example:
#   global_level = TRACE QUERY
```

Debugging – Wie?

Im Programm-Code:

```
$::lxdebug->message(LxDEBUG::INFO, "bb: bla");

$::lxdebug->message(0, "bb: bla");

$::lxdebug->warn("bb: bla");
```

Debugging – Wie?

```
$::lxdebug->dump(0, "bb: form", $::form);


my @my_array = ('a', 'b', 'c');
$::lxdebug->dump(0, "bb: ma", \@my_array);


$::lxdebug->dump(0, "bb: mc", \%::myconfig);
```

Debugging – Wie?

```
my $c = SL::DB::Manager::Customer->get_first;
$::lxdebug->dump(0, "bb: customer", $c);

$::lxdebug->dump(0, "bb: c-tree", $c->as_tree);
```

Debugging – Wie?

→ global_level = TRACE

```
sub my_sub {
  $::lxdebug->enter_sub();
  …
  $::lxdebug->leave_sub();
}
```

Debugging – Timing

```
global_level = REQUEST_TIMER
```

→ gibt die Zeit für den Request am Ende dessen aus.

```
$::lxdebug->log_time();
$::lxdebug->log_time("bb: die Zeit");
```

→ gibt die Zeit seit Start des Requests aus.

Debugging – SQL-Queries

```
global_level = QUERY
```

→ gibt die „alten“ SQL-Queries aus

→ nicht aber die über ROSE generierten

```
dbix_log4perl = 1
```

→ gibt auch ROSE-Queries aus

→ braucht DBIx::Log4Perl

Debugging – watch me

```
watch_form = 1
# $form->{"Watchdog::<variable>"} = 1;
```

→ Hash-Einträge („Variablen“) in $::form beobachten

```
# Beispiel: wo wird der Kunde gesetzt / geändert?
# also: $::form->{customer}

$::form->{"Watchdog::customer"} = 1;
```

Debugging – Informationen

→ Entwickler-Doku

(pdf oder <meine_kivi>/doc/html/04.html)

→ config/kivitendo.conf.default

→ perldoc SL/LXDebug.pm

Debugging – hands on

Mit TRACE und QUERY den Mahnungsfehler finden

(siehe http://redmine.kivitendo-premium.de/issues/54)