

Zero-Cost (Non-)Debug



- Problem:
 - High-Performance Proxies in Perl
 - Extensive Verwendung von Debugstatements gefährlich, selbst wenn Debugging aus
- Ziel:
 - Debugstatements, die bei Bedarf zu aktivieren sind und nur dann Performance kosten



```
#XDEBUG debug('foo');
```

```
#XDEBUG debug('foo %d bar %s',  
#XDEBUG      $bla,$fasel);
```

- [+] kein Performanceverlust
- [-] schwer zu aktivieren
- [-] kein Syntaxcheck



```
use myDebug; # debug, $DEBUG
$DEBUG && debug('foo');
$DEBUG && debug('foo %d bar %s',
    $bla, $fasel);
if ($DEBUG) { ... }
```

- [*] geringer Performanceverlust
- [+] leicht zu aktivieren
- [+] Syntaxcheck



```
'NODEBUG' || debug('foo');
```

```
$ perl -MO=Deparse -e '  
  "NODEBUG" || debug("foo") '  
'???' ;  
-e syntax OK
```

- [+] kein Performanceverlust
- [*] leichter zu aktivieren:
s/NODEBUG//g
- [-] kein Syntaxcheck



```
use constant DEBUG => 0;  
DEBUG && debug('foo');
```

```
$ perl -MO=Deparse -e '  
    use constant DEBUG => 0;  
    DEBUG && debug("foo")'  
use constant ('DEBUG', 0);  
'???';  
-e syntax OK
```

- [+] kein Performanceverlust
- [*] leichter zu aktivieren
- [+] Syntaxcheck



```
$ perl -M0=Deparse -e '  
  use constant DEBUG => $ENV{XDEBUG};  
  DEBUG && debug("foo")'  
use constant ('DEBUG', $ENV{'XDEBUG'});  
'???';
```

```
$ XDEBUG=1 perl -M0=Deparse -e '...'  
use constant ('DEBUG', $ENV{'XDEBUG'});  
debug('foo');
```

[+] leicht zu aktivieren



```
package myDebug;
use base 'Exporter';
our @EXPORT = qw( debug DEBUG );

# use constant foo => 'bar'
# gleiches wie: *foo = sub(){'bar'}
*DEBUG = $ENV{XDEBUG} ? sub(){1} : sub(){0};

sub debug {
    DEBUG() or return;
    print STDERR "@_\n";
}
```



Zero-Cost Non-Debug per Module



```
package myDebug;

my $DEBUG;
sub import {
    $DEBUG ||= [ map { qr/^\$_$/ }
        split(/,/, $ENV{XDEBUG}) ];

    my $pkg = caller;
    no strict 'refs';
    *{"${pkg}::debug"} = \&debug;
    # DEBUG spezifisch zu caller
    *{"${pkg}::DEBUG"} =
        ( grep { $pkg =~ $_ } @$DEBUG )
        ? sub(){1} : sub(){0};
}
sub debug { ... }
```

```
package Foo;
use myDebug;
DEBUG && debug('foo')
```

```
package Bar;
use myDebug;
DEBUG && debug('bar')
```

```
package Fool;
use myDebug;
DEBUG && debug('fool');
```

```
XDEBUG=Foo.* perl...
debug('foo')
'???'
debug('fool');
```



ENDE

