Alogger
One ring to rule them all

Alexander Dergachev
Nov 03, 2011

A bit of history

Good old days

```
-ifdef(DEBUG).
-define(LOG, ...).
-endif.
```

It's like 80s in your code. Michael Jackson, anyone?

Mockering

Some attempts were made

- log4erl
- fastlog
- elog
- lager this one is younger than alogger

What's the point?

All logging frameworks looks like brothers.

Nothing like «killing feature».

Rule them all

“One Ring to rule them all, One Ring to find them,
One Ring to bring them all and in the darkness bind them”

– J. R. R. Tolkien

Spawnfest

During spawnfest the ring was forged.

- Modularity
- OTP compliance
- Pluggable backends
- Full runtime control
- Concept of «flows»
- Almost zero overhead
- SASL errors are handled
- Syntactic convenience
- Production usage

Modularity

You can choose any logging backend, even in runtime

- tty
- disk log
- syslog
- scribe

Just implement gen_alog behaviour

Log levels

- emergency
- alert
- critical
- error
- warning
- notice
- info
- debug

Flows

Flow is... flow.

flow = filters + direction + metadata

filters: on module, tag, error level

direction: any logging backend

metadata: priority

Fighting with overhead

Almost zero overhead

?LOG is translated to alog_if:log

alog_if:log is generated, compiled and reloaded in runtime automatically

Look ma, no overhead (when logging is disabled)

Making overhead even lower

Sometimes gathering log data is expensive

You can do it lazily — just pass fun to logger

Syntactic magic

Boring one

```
FooValue = foo(),
?DBG("FooValue: ~s", [FooValue]),
FooValue
```

Funnier one

```
FooValue = foo(),
?DBG({FooValue}),
FooValue.
```

Returning right from the macro

```
FooValue = foo(),
?DBG({FooValue}).
```

Using arbitrary expression inside of a macro

```
?DBG(["FooValue", foo()]).
```

Measuring time of execution

```
?DBG_TC(["FooValue", foo()]).
```

Obtaining debug info

?DBG({FooValue, {lazy, "BarValue", fun() -> ... end}).

Wrapping up

Present status

Authors use it in production

Future

- more testing
- more backends
- nif-based io_lib_format version with size limits
- time tracking using splot
- formatters

Authors

- Alexander Dergachev
- Artem Golovinskiy
- Dmitry Groshev
- Igor Karymov

Thank you!