Package 'ot'

May 26, 2020

Type Package

Title 'Open Tracing'

Version 0.2.0

Author Neal Fultz <nfultz@gmail.com>

Maintainer Neal Fultz <nfultz@gmail.com>

Description 'Open Tracing' <https://opentracing.io> allows developers to add instrumentation to their application code using interfaces that are vendor-agnostic. This is used to monitor services, triage failures and find performance bottlenecks, among other things. The 'ot' package has generic methods to be extended when implementing the specification for a specific vendor.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

NeedsCompilation no

Repository CRAN

Date/Publication 2020-05-26 16:10:02 UTC

R topics documented:

getMockTracer 2
getMsgTracer 2
getNoOpTracer 3
ot 3
setTags 3
startSpan 5

Index **6**

getMockTracer *A Mock Tracer implementation*

Description

This implements a Tracer and Span which stores all function arguments in an environment.

Usage

```
getMockTracer()
```

Value

a new tracer instance

Examples

```
z <- ot::getMockTracer()
```

getMsgTracer *A Message Tracer*

Description

This implements a tracer that delegates function calls to `message()`.

Usage

```
getMsgTracer()
```

Value

a new tracer instance

Examples

```
z <- ot::getMsgTracer()
```

getNoOpTracer *A NoOp Tracer implementation*

Description

The OpenTracing specification mandates a tracer implementation which does nothing.

Usage

```
getNoOpTracer()
```

Value

a tracer instance

Examples

```
z <- ot::getNoOpTracer()
```

ot *Open Tracing*

Description

Open Tracing is a specification for trace logging. This package provides generics for the required methods, and a minimal implementation for testing purposes.

See Also

https://opentracing.io/

setTags *Span Object Methods*

Description

These define the core methods required by the specification for using spans.

Usage

```
setTags(span, ...)

baggage(span, ...)

baggage(span, ...) <- value

getContext(span, ...)

otlog(span, ..., timestamp = Sys.time())

log(span, ..., timestamp = Sys.time())

finish(span, finishTime = Sys.time())
```

Arguments

span	a span object
...	defined by implementation
value	the baggage data
timestamp	a POSIXct timestamp for the beginning of a span
finishTime	a POSIXct timestamp for the end of a span

Value

the span, except for getContext which returns the span's parent context and baggage, which returns any baggage objects.

Note

Developers should implement the `otlog` method only for their spans - log is a generic method used by R for logarithms. `ot::log` is an alias for convenience only.

Examples

```
s <- ot::startSpan(ot::getNoOpTracer())
ot::setTags(s, foo=1)
ot::baggage(s) <- list(ctx=1)
ot::getContext(s)
ot::otlog(s, foo=1)
ot::log(s, bar=2)
ot::finish(s)
ot::baggage(s)
```

`startSpan` *Tracer methods*

Description

Tracer objects encapsulate the state of the logging system. startSpan creates a span, and inject and extract set metadata via sidechannels.

Usage

```
startSpan(tracer, name, ...)

inject(tracer, contextObj, format, carrier)

extract(tracer, format, carrier)
```

Arguments

`tracer`	the tracing implementation
`name`	the name of the span
`...`	left to implementation
`contextObj`	a span or span context
`format`	One of the OpenTracing format values
`carrier`	A corresponding carrier object

Examples

```
z <- ot::getNoOpTracer()
ot::startSpan(z)
ot::inject(z, list("User-Agent"="R"), "HTTP_HEADERS", NULL)
ot::extract(z, "HTTP_HEADERS", NULL)
```

Index

baggage *(setTags)*, 3
baggage<- *(setTags)*, 3

extract *(startSpan)*, 5

finish *(setTags)*, 3

getContext *(setTags)*, 3
getMockTracer, 2
getMsgTracer, 2
getNoOpTracer, 3

inject *(startSpan)*, 5

log *(setTags)*, 3

ot, 3
otlog *(setTags)*, 3

setTags, 3
startSpan, 5