Robotics Service Bus - Feature #2311
Allow negative scope filtering in the logger

06/02/2015 02:58 PM - N. Köster

Status:	Rejected	**Start date:**	06/02/2015
Priority:	Normal	**Due date:**	
Assignee:	J. Moringen	**% Done:**	0%
Category:	Common Lisp Tools	**Estimated time:**	0.00 hour
Target version:	rsb-0.13		

Description

According to the logger documentation, it is only possible to positively filter scopes, e.g.:

```
tools0.12 logger --on-error=continue --filter 'scope "/some_important_scope/"' --style monitor/timeline
```

gives you all scopes that match the scope " /**some_important_scope**/ "

However, it seems that it is not possible to negate this filter in any way. Something I'd expect possible is (or however the correct syntax would be ;)):

```
tools0.12 logger --on-error=continue --filter 'scope :scope "/some_unimportant_scope/" :negate' --style monitor/timeline
```

which would **filter out** the named scope (and possibly all its subscopes...).

Related issues:

Duplicates Robotics Service Bus - Enhancement # 1777: Add commandline options...	**In Progress**	**02/18/2014**

History

#1 - 06/02/2015 03:05 PM - J. Moringen

- Duplicates Enhancement #1777: Add commandline options to exclude scopes from logging added

#2 - 06/02/2015 03:17 PM - J. Moringen

It is already possible to construct composite filter expressions involving not, and, or. There is however no convenient syntax[1] for writing these down.

In the meantime, use

```
$ rsb-tools logger -f 'not :children #.(list (rsb.filter:filter :scope :scope "/foo" :exact? EXACT)'
```

where EXACT is either t or nil and controls whether to exclude sub-scopes.

[1] Well, I *have* implemented such a syntax, but it is not in master.

#3 - 06/02/2015 03:17 PM - J. Moringen

- Status changed from New to Rejected

#4 - 06/02/2015 06:25 PM - N. Köster

ah thanks! I did not find the existing issue in my search!

J. Moringen wrote:

> *In the meantime, use*
> *[...]*
>
> *where EXACT is either t or nil and controls whether to exclude sub-scopes.*

Unfortunately I am not quite sure how to use this? Would I have to replace EXACT? And with a literal 't' or 'nil'?

{{collapse(Here is a result in my example when I call this (I want hide all system/monitoring related scopes)...)

```
$ tools0.12 logger -f 'not :children #.(list (rsb.filter:filter :scope :scope "/system/monitoring/" :exact? EXACT)' --debug

Unhandled END-OF-FILE in thread #<SB-THREAD:THREAD "main thread" RUNNING
                                  {100A7CEF53}>:
  end of file on #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}>

Backtrace for: #<SB-THREAD:THREAD "main thread" RUNNING {100A7CEF53}>
0: ((LAMBDA NIL :IN SB-DEBUG::FUNCALL-WITH-DEBUG-IO-SYNTAX))
1: (SB-IMPL::CALL-WITH-SANE-IO-SYNTAX #<CLOSURE (LAMBDA NIL :IN SB-DEBUG::FUNCALL-WITH-DEBUG-IO-SYNTAX)
{100B11F4AB}>)
2: (SB-IMPL::%WITH-STANDARD-IO-SYNTAX #<CLOSURE (LAMBDA NIL :IN SB-DEBUG::FUNCALL-WITH-DEBUG-IO-SYNTAX)
{100B11F47B}>)
3: (PRINT-BACKTRACE :STREAM #<SYNONYM-STREAM :SYMBOL SB-SYS:*STDERR* {10001C52B3}> :START 0 :FROM
:INTERRUPTED-FRAME :COUNT NIL :PRINT-THREAD T :PRINT-FRAME-SOURCE NIL :METHOD-FRAME-STYLE NIL)
4: (SB-DEBUG::DEBUGGER-DISABLED-HOOK #<END-OF-FILE {100B11BED3}> #<unavailable argument>)
5: (SB-DEBUG::RUN-HOOK *INVOKE-DEBUGGER-HOOK* #<END-OF-FILE {100B11BED3}>)
6: (INVOKE-DEBUGGER #<END-OF-FILE {100B11BED3}>)
7: (UIOP/IMAGE:HANDLE-FATAL-CONDITION #<END-OF-FILE {100B11BED3}>)
8: (SIGNAL #<END-OF-FILE {100B11BED3}>)
9: (ERROR END-OF-FILE :STREAM #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}>)
10: (SB-IMPL::STRING-INCH #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> T NIL)
11: (SB-IMPL::FLUSH-WHITESPACE #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}>)
12: (SB-IMPL::READ-LIST #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> #<unavailable argument>)
13: (SB-IMPL::READ-MAYBE-NOTHING #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> #\()
14: (SB-IMPL::%READ-PRESERVING-WHITESPACE #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> T (NIL) T)
15: (READ #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> T NIL T)
16: (SB-IMPL::SHARP-DOT #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> #\. NIL)
17: (SB-IMPL::READ-MAYBE-NOTHING #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> #\#)
18: (SB-IMPL::%READ-PRESERVING-WHITESPACE #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> NIL (NIL) T)
19: (SB-IMPL::%READ-PRESERVING-WHITESPACE #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> NIL (NIL) NIL)
20: (READ #<SB-IMPL::STRING-INPUT-STREAM {100B11BC63}> NIL #:EOF187 NIL)
21: (RSB.COMMON:PARSE-INSTANTIATION-SPEC "not :children #.(list (rsb.filter:filter :scope :scope \"/system/monitoring/\" :exact?
EXACT)")
22: (RSB.TOOLS.LOGGER:MAIN #P"tools0.12" ("-f" "not :children #.(list (rsb.filter:filter :scope :scope \"/system/monitoring/\" :exact? EXACT)"
"--debug"))
23: (RSB.TOOLS.MAIN::MAIN/COMMAND #P"tools0.12" "logger" ("-f" "not :children #.(list (rsb.filter:filter :scope :scope \"/system/monitoring/\"
:exact? EXACT)" "--debug"))
24: (RSB.TOOLS.MAIN:MAIN)
25: ((LAMBDA NIL :IN UIOP/IMAGE:RESTORE-IMAGE))
```

26: (UIOP/IMAGE:CALL-WITH-FATAL-CONDITION-HANDLER #<CLOSURE (LAMBDA NIL :IN UIOP/IMAGE:RESTORE-IMAGE) {100A7D7F7B}>)
27: ((FLET #:WITHOUT-INTERRUPTS-BODY-89 :IN SAVE-LISP-AND-DIE))
28: ((LABELS SB-IMPL::RESTART-LISP :IN SAVE-LISP-AND-DIE))

unhandled condition in --disable-debugger mode, quitting

}}

#5 - 06/02/2015 07:10 PM - J. Moringen

N. Köster wrote:

> *Unfortunately I am not quite sure how to use this? Would I have to replace EXACT? And with a literal 't' or 'nil'?*

Yes, literal t (true) or nil (false).

> *$ tools0.12 logger -f 'not :children #.(list (rsb.filter:filter :scope :scope "/system/monitoring/" :exact? EXACT)' --debug*
> *^ missing ")"*
> *^ t or nil*