



Tools

Copyright © 1997-2019 Ericsson AB. All Rights Reserved.
Tools 3.2
July 11, 2019

Copyright © 1997-2019 Ericsson AB. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0> Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Ericsson AB. All Rights Reserved..

July 11, 2019

1.1 cover

```
-module(channel).
-behaviour(gen_server).

-export([start_link/0,stop/0]).
-export([alloc/0,free/1]). % client interface
-export([init/1,handle_call/3,terminate/2]). % callback functions

start_link() ->
    gen_server:start_link({local,channel},channel,[],[]).

stop() ->
    gen_server:call(channel,stop).

%%%Client interface functions-----

alloc() ->
    gen_server:call(channel,alloc).

free(Channel) ->
    gen_server:call(channel,{free,Channel}).

%%%gen_server callback functions-----

init(_Arg) ->
    {ok,channels()}.

handle_call(stop,Client,Channels) ->
    {stop,normal,ok,Channels};

handle_call(alloc,Client,Channels) ->
    {Ch,Channels2} = alloc(Channels),
    {reply,{ok,Ch},Channels2};

handle_call({free,Channel},Client,Channels) ->
    Channels2 = free(Channel,Channels),
    {reply,ok,Channels2}.

terminate(_Reason,Channels) ->
    ok.

%%%Internal functions-----

channels() ->
    [ch1,ch2,ch3].

alloc([Channel|Channels]) ->
    {Channel,Channels};
alloc([]) ->
    false.

free(Channel,Channels) ->
    [Channel|Channels].
```

The test case is implemented as follows:

```
-module(test).
-export([s/0]).

s() ->
    {ok,Pid} = channel:start_link(),
    {ok,Ch1} = channel:alloc(),
    ok = channel:free(Ch1),
    ok = channel:stop().
```



```
5> cover:analyse(channel,coverage,function).
{ok,[{{channel,start_link,0},{1,0}},
      {{channel,stop,0},{1,0}},
      {{channel,alloc,0},{1,0}},
      {{channel,free,1},{1,0}},
      {{channel,init,1},{1,0}},
      {{channel,handle_call,3},{5,0}},
      {{channel,terminate,2},{1,0}},
      {{channel,channels,0},{1,0}},
      {{channel,alloc,1},{1,1}},
      {{channel,free,2},{1,0}}]}
```

For `channel`, the result shows that the uncovered line is in the function `channel:alloc/1`.

If the analysis is made on clause level, the result is given as a list of tuples `{Clause, {Cov,NotCov}}`, one for each function clause in the module. A clause is specified by its module name, function name, arity and position within the function definition:

```
6> cover:analyse(channel,coverage,clause).
{ok,[{{channel,start_link,0,1},{1,0}},
      {{channel,stop,0,1},{1,0}},
      {{channel,alloc,0,1},{1,0}},
      {{channel,free,1,1},{1,0}},
      {{channel,init,1,1},{1,0}},
      {{channel,handle_call,3,1},{1,0}},
      {{channel,handle_call,3,2},{2,0}},
      {{channel,handle_call,3,3},{2,0}},
      {{channel,terminate,2,1},{1,0}},
      {{channel,channels,0,1},{1,0}},
      {{channel,alloc,1,1},{1,0}},
      {{channel,alloc,1,2},{0,1}},
      {{channel,free,2,1},{1,0}}]}
```

For `channel`, the result shows that the uncovered line is in the second clause of `channel:alloc/1`.

Finally, if the analysis is made on line level, the result is given as a list of tuples `{Line, {Cov,NotCov}}`, one for each executable line in the source code. A line is specified by its module name and line number.

```
7> cover:analyse(channel,coverage,line).
{ok,[{{channel,9},{1,0}},
      {{channel,12},{1,0}},
      {{channel,17},{1,0}},
      {{channel,20},{1,0}},
      {{channel,25},{1,0}},
      {{channel,28},{1,0}},
      {{channel,31},{1,0}},
      {{channel,32},{1,0}},
      {{channel,35},{1,0}},
      {{channel,36},{1,0}},
      {{channel,39},{1,0}},
      {{channel,44},{1,0}},
      {{channel,47},{1,0}},
      {{channel,49},{0,1}},
      {{channel,52},{1,0}}]}
```

For `channel`, the result shows that the uncovered line is line number 49.

instrument

histogram_width

The number of intervals in the free block size histograms. Defaults to 14.

Example:

```
> instrument:carriers({ histogram_start => 512, histogram_width => 8 }).
{ok,{512,
  [{ll_alloc,1048576,0,1048344,71,false,{0,0,0,0,0,0,0,0}},
   {binary_alloc,1048576,0,324640,13,false,{3,0,0,1,0,0,0,2}},
   {eheap_alloc,2097152,0,1037200,45,false,{2,1,1,3,4,3,2,2}},
   {fix_alloc,32768,0,29544,82,false,{22,0,0,0,0,0,0,0}},
   {...}|...]}}
```

See Also

erts_alloc(3), *erl(1)*

SEE ALSO

- Richard M. Stallman. GNU Emacs Manual, chapter "Editing Programs", section "Tag Tables". Free Software Foundation, 1995.
- Anders Lindgren. The Erlang editing mode for Emacs. Ericsson, 1998.

