


 [pipo02mix](#) / **docker-ejabberd** Public

forked from [roemhild/docker-ejabberd](#)


<> **Code**


 Pull requests

 Actions

 Projects

 Security

 Insights

 master ▾

...

docker-ejabberd / [mod_cobrowser](#) / [src](#) / **mod_cobrowser.erl**



Fernando Ripoll feat(docker): Have all needed to run on docker the new eja...

 **History**

 0 contributors

92 lines (77 sloc) | 3.4 KB

...

```

1  %%%-----
2  %%% File      : mod_cobrowser.erl
3  %%% Author   : pipo02mix
4  %%% Purpose  : Post availability to API endpoint
5  %%% Created  : 3 April 2017
6  %%% Id       : $Id: mod_cobrowser.erl 1034 2017-04-01 19:04:17Z pipo02mix $
7  %%%-----
8
9  -module(mod_cobrowser).
10 -author('fernando@cobrowser.net').
11
12 -behaviour(gen_mod).
13
14 %% Required by ?DEBUG macros
15 -include("logger.hrl").
16 -include("xmpp.hrl").
17
18 %% gen_mod API callbacks
19 -export([start/2, stop/1, on_user_send_packet/1, on_disconnect/3, send_availability/
20
21 start(Host, _Opts) ->
22     ?INFO_MSG("mod_cobrowser starting", []),
23     inets:start(),
24     ejabberd_hooks:add(user_send_packet, Host, ?MODULE, on_user_send_packet, 50),
25     ejabberd_hooks:add(sm_remove_connection_hook, Host, ?MODULE, on_disconnect, 50),
26     ?INFO_MSG("mod_cobrowser hooks attached", []),
27     ok.
28
29 stop(Host) ->
30     ?INFO_MSG("mod_cobrowser stopping", []),

```

```

31
32     ejabberd_hooks:delete(user_send_packet, Host, ?MODULE, on_user_send_packet, 50),
33     ejabberd_hooks:delete(sm_remove_connection_hook, Host, ?MODULE, on_disconnect, 5
34     ok.
35
36 -spec on_user_send_packet({stanza(), ejabberd_c2s:state()}) -> {stanza(), ejabberd_c
37 on_user_send_packet({#presence{
38         from = #jid{lresource = <<">>} = From,
39         show = Show,
40         type = unavailable = Type} = Pkt, State} ) ->
41
42     Jid = binary_to_list(jlib:jid_to_string(From)),
43     BareJid = string:sub_string(Jid,1,string:str(Jid,"/")-1),
44     send_availability(BareJid, Type, Show),
45     {Pkt, State};
46 on_user_send_packet({#presence{
47         from = From,
48         show = Show,
49         type = available = Type} = Pkt, State} ) ->
50
51     Jid = binary_to_list(jlib:jid_to_string(From)),
52     BareJid = string:sub_string(Jid,1,string:str(Jid,"/")-1),
53     send_availability(BareJid, Type, Show),
54     {Pkt, State};
55 on_user_send_packet(Acc) ->
56     Acc.
57
58 on_disconnect(Sid, Jid, Info ) ->
59     StrJid = binary_to_list(jlib:jid_to_string(Jid)),
60     BareJid = string:sub_string(StrJid,1,string:str(StrJid,"/")-1),
61     ?DEBUG("(mod_cobrowser)onDisconnect: ~p, ~p, ~p", [ Sid, BareJid, Info]),
62     send_availability(BareJid, unavailable, undefined),
63
64     ok.
65
66 send_availability(Jid, Type, Show) ->
67     APIHost = getenv("NGINX_INTERNAL_SERVICE_HOST", "nginx-internal.default.svc.cl
68     APIEndpoint = "http://" ++ APIHost ++ "/api/app.php/internal/availability/user
69     ShowString = lists:flatten(io_lib:format("~p", [ Show])),
70     TypeString = lists:flatten(io_lib:format("~p", [ Type])),
71     ?DEBUG("sending packet: ~p type: ~p show: ~p api: ~p", [ Jid, Type, Show, APIE
72     URL = "jid=" ++ Jid ++ "&type=" ++ TypeString ++ "&show=" ++ ShowString,
73     R = httpc:request(post, {
74         APIEndpoint,
75         [],
76         "application/x-www-form-urlencoded",
77         URL}, [], []),
78     {ok, {{ "HTTP/1.1", ReturnCode, _}, _, _}} = R,
79     ?DEBUG("API request made with result -> ~p ", [ ReturnCode]),
80     ReturnCode.
81

```

```
82 -spec depends(binary(), gen_mod:opts()) -> [{module(), hard | soft}].
83 depends(_Host, _Opts) ->
84     [].
85
86 getenv(VarName, DefaultValue) ->
87     case os:getenv(VarName) of
88         false ->
89             DefaultValue;
90         Value ->
91             Value
92     end.
```