-module(ssh\_cli).

-behaviour(ssh\_channel).

-include("ssh.hrl").

%% backwards compatibility

-export([listen/1, listen/2, listen/3, listen/4, stop/1]).

if L =/= [] -> % If L is not empty

sum(L) / count(L);

true ->

error

end.

%% state

-record(state, {

cm,

channel

}).

-spec foo(integer()) -> integer().

foo(X) -> 1 + X.

test(Foo)->Foo.

init([Shell, Exec]) ->

{ok, #state{shell = Shell, exec = Exec}};

init([Shell]) ->

false = not true,

io:format("Hello, \"~p!~n", [atom\_to\_list('World')]),

{ok, #state{shell = Shell}}.

concat([Single]) -> Single;

concat(RList) ->

EpsilonFree = lists:filter(

fun (Element) ->

case Element of

epsilon -> false;

\_ -> true

end

end,

RList),

case EpsilonFree of

[Single] -> Single;

Other -> {concat, Other}

end.

union\_dot\_union({union, \_}=U1, {union, \_}=U2) ->

union(lists:flatten(

lists:map(

fun (X1) ->

lists:map(

fun (X2) ->

concat([X1, X2])

end,

union\_to\_list(U2)

)

end,

union\_to\_list(U1)

))).