- Erlang - sehr praktische Einführung

EH2008 - 21.3.2008

Ben Fuhrmannek <ben@fuhrmannek.de>

Overview

- History/Facts - Worum geht es überhaupt?
- Console - viele Beispiele
- Functions & Modules
- Concurrency - Prozesse & Nachrichten
- Example - Prozesse & Nachichten im Detail
- Workshop-Teil

History/Facts

- Zielvorgabe: Sprache/Plattform für Telekommunikation
- Concurrency, Robustness, Distribution
- Open Source seit 1998
- bekannte Anwendungen: ejabberd (jabber.org, jabber.ccc.de), tsung, yaws, Amazon SimpleDB

Jump in (interactive)

Console [numbers, atoms, tuples]:

```
$ erl
Erlang (BEAM) emulator version 5.6 [source] [smp:2] [async-
threads:0] [kernel-poll:false]

Eshell V5.6  (abort with ^G)
1> 5.
5
2> 23 + 23#42.
117
3> 23, 42.
42
4> abc.
abc
5> 'a bc'.
'a bc'
6> {1, 2}.
{1,2}
7>
```

atom

tuple (fixed length array)

Console [lists]

```
7> [1,2,3].
[1,2,3]
8> "abc".
"abc"
9> [97,98,99].
"abc"
10> [1,2,3] ++ [4,5,6].
[1,2,3,4,5,6]
11> [1,2,3 | [4,5,6]].
[1,2,3,4,5,6]
12> ["def" | "abc"].
["def",97,98,99]
13> [X * 2 || X <- [1,2,3]].
[2,4,6]
14>
14> list_to_integer("23").
23
```

lists

list comprehension

Console [Vars, matching]

Variables (^[A-Z])

single-assignment!

```
31> A = 1.
1
32> A = 2.
** exception error: no match of right hand side value 2
33> A = 1.
1
34> A =:= 1.0.
false
35> A == 1.0.
true
36>
36> [B] = "b".
"b"
37> [_,_,C|_] = "fnord", C.
111
```

Console <<bit syntax>>

```
38> <<2#01000000000100000000100:24/big>>.
<<64,16,4>>
39> <<2#01000000000100000000100:24/little>>.
<<4,16,64>>
40> <<D:24/unsigned-integer-big>> = <<1,2,3>>, D.
** exception error: no match of right hand side value <<1,2,3>>
41> f().
ok
42> <<D:24/unsigned-integer-big>> = <<1,2,3>>, D.
66051
```

Functions & Modules

file: m.erl

```
-module(m).
-export([fac/1]).

fac(X) when X >= 1 ->
  fac(X, 1).

fac(1, M) ->
  M;
fac(X, M) ->
  fac(X-1, M * X).
```

guard

tail-recursion

Functions & Modules

```
$ erl
Erlang (BEAM) emulator version 5.6 [source] [smp:2] [async-
threads:0] [kernel-poll:false]

Eshell V5.6  (abort with ^G)
1> c(m).
{ok,m}
2> m:fac(30).
265252859812191058636308480000000
```

Concurrency: about processes and messages

"The world is parallel.

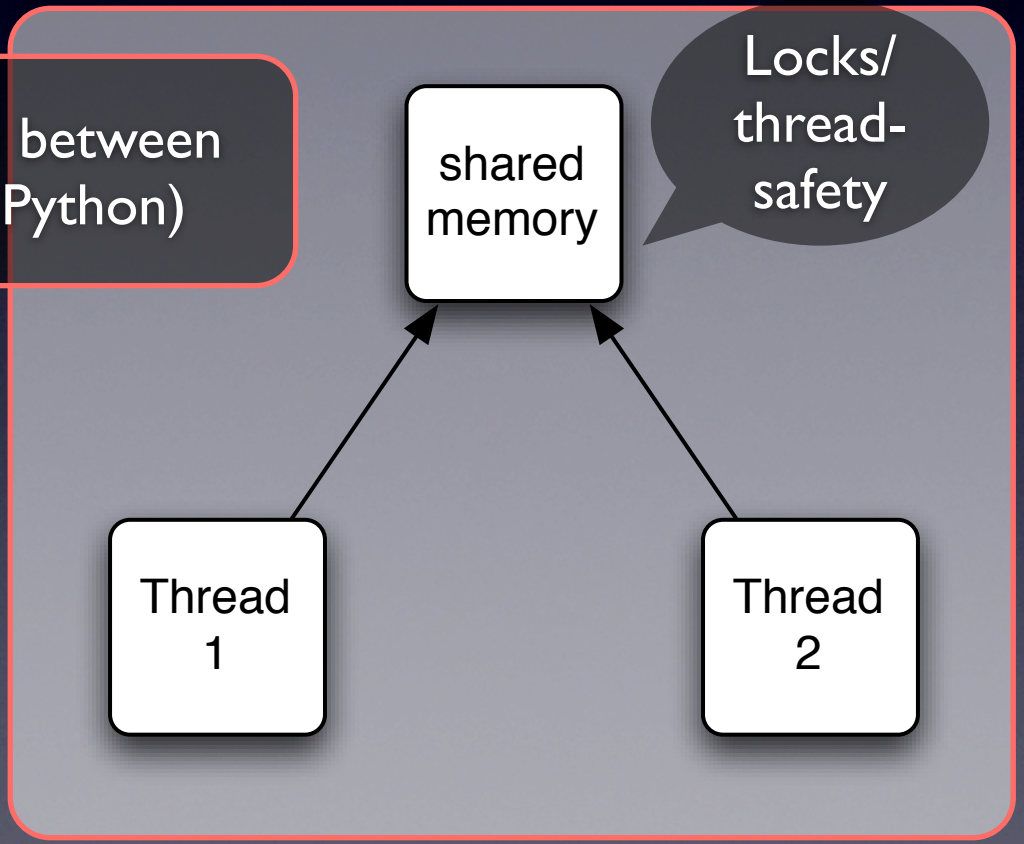
If we want to write programs that behave as other objects behave in the real world, then these programs will have a concurrent structure. Use a language that was designed for writing concurrent applications, and development becomes a lot easier.

Erlang programs model how we think and interact."

- Joe Armstrong

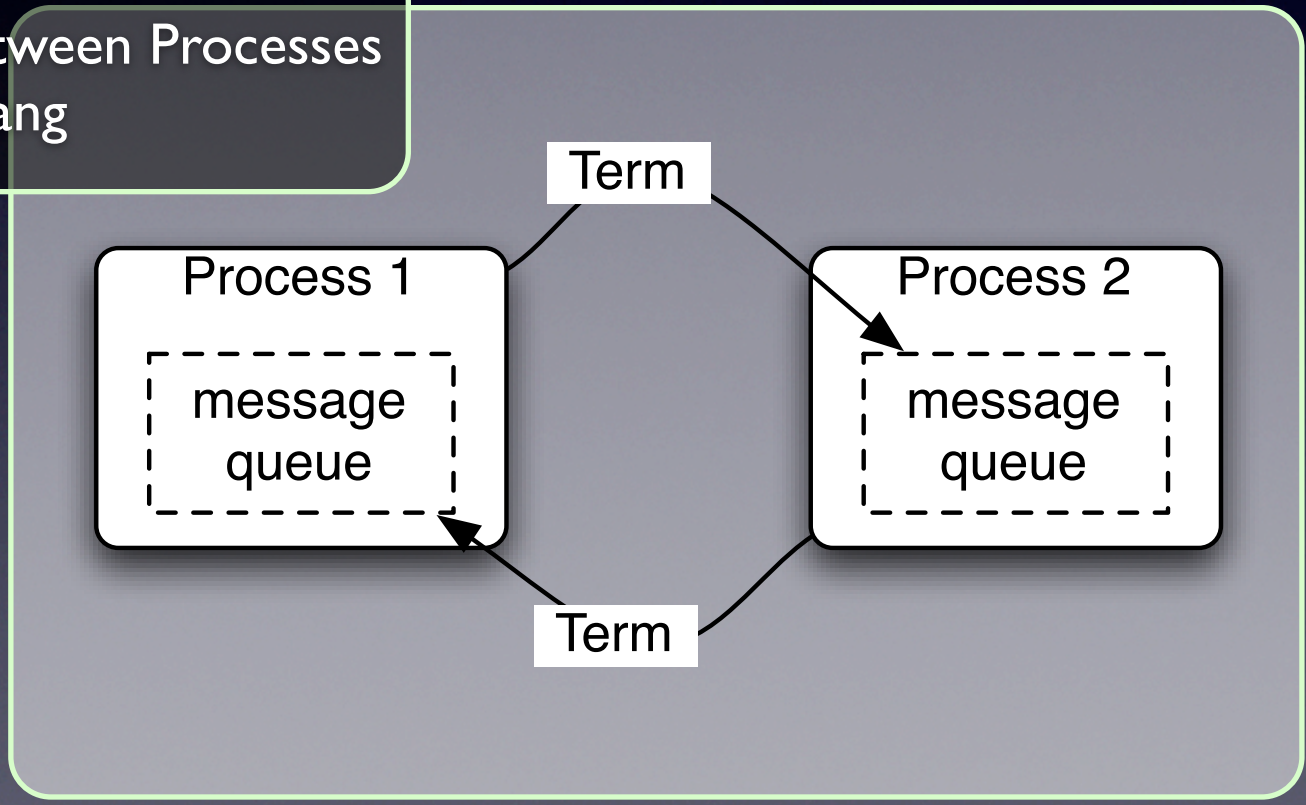
Concurrency: about processes and messages

common communication between Threads (e.g. in Java or Python)



Concurrency: about processes and messages

communication between Processes in Erlang



Processes, Messages (Code)

```
-module(foo).
-export([
    bar/0,
    test/0
    ]).

bar() ->
    receive
        quit -> ok;
        X ->
            io:format("received ~p~n", [X]),
            bar()
    end.

test() ->
    P = spawn(fun bar/0),
    P ! test,
    P ! {this, is, "a", 7337},
    P ! quit.
```

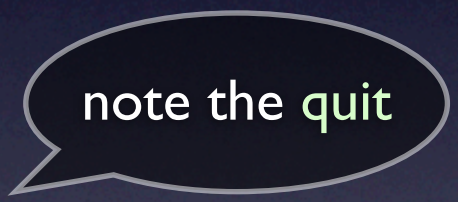
receive

new process

send !

Processes, Messages (Compile & Run)

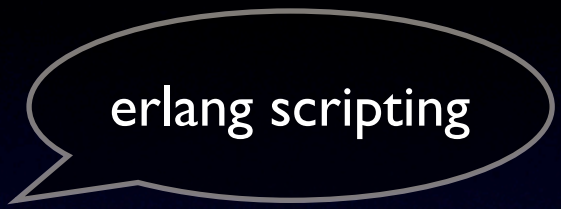
```
5> c(foo).
{ok,foo}
6> foo:test().
received test
quit
received {this,is,"a",7337}
```



Example: URL rewriter

```
#!/usr/bin/env escript

main(_) ->
     error_logger:tty(false),
     Sender = spawn(fun sender/0),
     receiver(Sender).

sender() ->
     receive
          quit -> ok;
          {put, S} ->
               io:format("~s", [S]),
               sender()
     end.

receiver(Sender) ->
     case io:get_line('') of
          eof ->
               Sender ! quit,
               halt(0);
          String ->
               spawn(fun() -> worker(Sender, String) end),
               receiver(Sender)
     end.

worker(Sender, String) ->
     [ID, URL, _Client, _User, _Method, _URLGroup|_] = string:tokens(String, " "),

     %% this computation takes really long
     timer:sleep(5000),

     NewURL = case URL of
          [$h,$t,$t,$p,$s,$:,$/,$/|_] -> URL;
          _ -> "about:insecure"
     end,

     Sender ! {put, lists:flatten(io_lib:format("~s ~s~n", [ID, NewURL]))}.
```

Ex. step by step [1]




```
#!/usr/bin/env escript

main(_) ->
    error_logger:tty(false),
    Sender = spawn(fun sender/0),
    receiver(Sender).
```

Ex. step by step [2]

```
sender() ->
   receive
      quit -> ok;
      {put, S} ->
         io:format("~s", [S]),
         sender()
   end.

receiver(Sender) ->
   case io:get_line('') of
      eof ->
         Sender ! quit,
         halt(0);
      String ->
         spawn(fun() -> worker(Sender, String) end),
         receiver(Sender)
   end.
```

Ex. step by step [3]

```
worker(Sender, String) ->
    [ID, URL, _Client, _User, _Method, _URLGroup|_] =
        string:tokens(String, " "),

    %% this computation takes really long
    timer:sleep(5000),

    NewURL = case URL of
        [$h,$t,$t,$p,$s,$:,$/,$/|_] -> URL;
        _ -> "about:insecure"
    end,

    Sender ! {put,
        lists:flatten(io_lib:format("~s ~s~n", [ID, NewURL]))}.
```

left out

- OTP - Open Telecom Platform
- mnesia database
- ... a lot more

Read on

- Web: official Erlang documentation: http://www.erlang.org/doc/
- Book: Programming Erlang - Software for a concurrent world - Author: Joe Armstrong
- Community Site: http://trapexit.org/
- Manual pages: `erl -man <foo>`
  e.g. `erl -man io` or `erl -man erl`

Workshop-Teil

a. Install erlang
b. Replay the console examples
c. ‘the task’.