



Erlang

Desenvolupant sistemes escalables massius en temps real

Josep de Cid

January 4, 2018



- 1 Erlang/OTP
- 2 Context històric
- 3 Qui l'utilitza?
- 4 Característiques
- 5 Exemples

Què és Erlang?

Erlang/OTP?



Erlang

- Llenguatge de programació funcional i concurrent.
- Sistema d'execució mitjançant una màquina virtual.

OTP (Open Telecom Platform)

- Conjunt biblioteques estàndard de suport al llenguatge.
- Proporciona estructures de dades, sockets, control d'errors, protocols de xarxa, interfícies per llenguatges externs (C, C++, Java, TCP...)

Degut a la normalització de l'ús conjunt d'Erlang i OTP, el llenguatge també és coneix com a Erlang/OTP.



- 1 Erlang/OTP
- 2 Context històric
- 3 Qui l'utilitza?
- 4 Característiques
- 5 Exemples

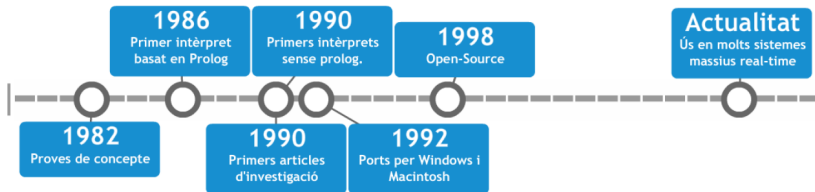
Context històric

Origen i motius



Per què Erlang?

- Complexitat en els sistemes Ericsson (+20 llenguatges).
- Facilitar programació concurrent.
- Simplificar la distribució en xarxa.
- Optimitzar tractament d'errors.





- 1 Erlang/OTP
- 2 Context històric
- 3 Qui l'utilitza?**
- 4 Característiques
- 5 Exemples

Qui l'utilitza?

Productes a producció amb Erlang





- 1 Erlang/OTP
- 2 Context històric
- 3 Qui l'utilitza?
- 4 Característiques**
- 5 Exemples



Multi-paradigma

- Propòsit general.
- Funcional.
- Concurrent.
- Distribuït.

Interpretat

- Compilat a Bytecode.
- Interpretat per la màquina virtual BEAM.
- Compiladors natius externs com HiPE.

Característiques

Tipus i similars

Sistema de tipus

- Tipatge dinàmic.
- Fortament tipat.

Tipus

- Integer, Float, Atom, Pid, Fun, Reference, Binary, Port.
- Tuples, List, Map.
- String, Record.

Llenguatges similars

- Sintàcticament: Prolog, LISP, Smalltalk o PLEX.
- Conceptualment: occam.



- 1 Erlang/OTP
- 2 Context històric
- 3 Qui l'utilitza?
- 4 Característiques
- 5 Exemples**

Exemples

Bàsic seqüencial



```
-module(listLib).  
-export([append/2]).  
-export([sort/1]).
```

```
append([], E) -> E;  
append([H|T], E) -> [H|append(T, E)].
```

```
sort([]) -> [];  
sort([Pivot | T]) -> sort([X || X <- T, X < Pivot])  
    ++ [Pivot] ++ sort([X || X <- T, X >= Pivot]).
```

Exemples

Bàsic seqüencial



```
1> c(listLib).  
{ok,listLib}  
2> listLib:append([2,0,1], [8]).  
[2,0,1,8]  
3> listLib:append("Jose", "p").  
"Josep"  
4> listLib:sort([1,2,1,5,2,3,7,4,-2]).  
[-2,1,1,2,2,3,4,5,7]
```

Exemples

Bàsic seqüencial



```
5> % Adder
5> Adder = fun(X) -> fun(Y) -> X + Y end end.
#Fun<erl_eval.6.99386804>
6> % Bind Fun
6> G = Adder(10).
#Fun<erl_eval.6.99386804>
7> % Fun result
7> G(5).
15
```

Exemples

Concurrencia entre processos

```
-module(pingPong).  
-export([ping/2, pong/0, start/0])
```

```
ping(0, Pong_PID) ->  
    Pong_PID ! finished, io:format("ping finished.\n", []);  
ping(N, Pong_PID) ->  
    Pong_PID ! {ping, self()},  
    receive pong -> io:format("ping received pong.\n", []) end,  
    ping(N-1, Pong_PID).
```

```
pong() ->  
    receive  
        finished -> io:format("pong finished.\n", []);  
        {ping, Ping_PID} ->  
            io:format("pong received ping.\n", []),  
            Ping_PID ! pong, pong()  
    end.
```

Exemples

Concurrencia entre processos



```
start() ->
    Pong_PID = spawn(pingPong, pong, []),
    spawn(pingPong, ping, [3, Pong_PID]).
```

```
1> c(pingPong).
{ok,pingPong}
2> pingPong:start().
pong received ping.
ping received pong.
pong received ping.
ping received pong.
pong received ping.
ping received pong.
ping finished.
pong finished.
```




```
...  
true = net_kernel:connect_node(Node),  
  
Pid1 = spawn(Node, Fun),  
Pid2 = spawn(Node, Module, Func, Argv),  
  
true = is_process_alive(Pid1),  
...
```






Exemples

Control d'errors



```
case (catch foo(A, B)) of
  {abnormal_case1, Y} ->
    ...
  {'EXIT', Opps} ->
    ...
  Val ->
    ...
end,
...
foo(A, B) ->
  ...
  throw({abnormal_case1, ...})
```



-  Erlang/OTP 20.2 Documentation [03/01/2018]
<http://erlang.org/doc/>
-  Learn You Some Erlang [27/12/2017]
<http://learnyousomeerlang.com>
-  Erlang (programming language) [27/12/2017]
<https://en.wikipedia.org/wiki/Erlang>
-  Cesarini, Francesco; Thompson, Simon.
Erlang Programming. O'Reilly Media; June 2009
-  Armstrong, Joe. *Making reliable distributed systems in the presence of software errors*. Ph.D, The Royal Institute of Technology, Stockholm, Sweden; November 20, 2003