Wstęp do Erlanga

Mateusz Lenik

Erlang...

Erlang...

- ...powstał latach 80 ubiegłego wieku...

Erlang...

- ...powstał latach 80 ubiegłego wieku...
- ...to funkcyjny język programowania...

Erlang...

- ...powstał latach 80 ubiegłego wieku...
- ...to funkcyjny język programowania...
- ...jest Open Source

Erlang...

- dynamicznie typowanie

Erlang...

- dynamicznie typowanie
- pojedyncze przypisanie

Erlang...

- dynamicznie typowanie
- pojedyncze przypisanie
- zachłanna ewaluacja

Erlang...

- dynamicznie typowanie
- pojedyncze przypisanie
- zachłanna ewaluacja
- brak pętli

Erlang pozwala...

- ...tworzyć rozproszone systemy...

Erlang pozwala...

- ...tworzyć rozproszone systemy...
- ...o wysokiej dostępności...

Erlang pozwala...

- ...tworzyć rozproszone systemy...
- ...o wysokiej dostępności...
- ...i **niezawodności**

Zastosowania...

- systemy telekomunikacyjne (switche, itp.)

Zastosowania...

- systemy telekomunikacyjne (switche, itp.)
- bazy danych (CouchDB, Riak)

Zastosowania...

- systemy telekomunikacyjne (switche, itp.)
- bazy danych (CouchDB, Riak)
- serwery (Yaws, WebMachine, ejabberd)

WebMachine

WebMachine

- framework dla aplikacji internetowych

WebMachine

- framework dla aplikacji internetowych
- wszystko jest zasobem

WebMachine

- framework dla aplikacji internetowych
- wszystko jest zasobem
- implementacja w Erlangu lub Ruby

WebMachine

- framework dla aplikacji internetowych
- wszystko jest zasobem
- implementacja w Erlangu lub Ruby
- oparta na maszynie stanowej

WebMachine

- framework d
- WS
-
- op

ych

lub Ruby

ynie stanowej

HTTP done The Right Way™

Hello, Erlang!

```
-module(hello).
-export([greet/0]).

greet() ->
  io:format("Hello, Erlang!~n").
```

Hello, Erlang!

```
$ erl
Eshell V5.9  (abort with ^G)
1> c(hello).
{ok,hello}
2> hello:greet().
Hello, Erlang!
ok
3>
```

Hello, Erlang!

```
-module(hello).
-export([greet/0]).

greet() ->
  io:format("Hello, Erlang!~n").
```

Hello, Erlang!

```
-module(hello).
-export([greet/0]).

greet() ->
  io:format("Hello, Erlang!~n").
```

Hello, Erlang!

```
-module(hello).
-export([greet/0]).

greet() ->
  io:format("Hello, Erlang!~n").
```

Hello, Erlang!

```
-module(hello).
-export([greet/0]).
-import(io, [format/1]).

greet() ->
    format("Hello, Erlang!~n").
```

Silnia

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

fac(N) when N > 0 -> N*fac(N-1);
fac(0) -> 1.
```

Silnia

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

fac(N) when N > 0 -> N*fac(N-1);
fac(0) -> 1.
```

Silnia

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

fac(N) when N > 0 -> N*fac(N-1);
fac(0) -> 1.
```

Silnia

```
-module(math).
-export([fac/1, fac_tail/1]).
...

fac_tail(N) -> fac_tail(N, 1).
fac_tail(0, Acc) -> Acc;
fac_tail(N, Acc) when N > 0 ->
  fac_tail(N-1, N*Acc).
```

Odwracanie list

```
-module(list).
-export([rev/1]).

rev([]) -> [];
rev([First|Rest]) ->
  rev(Rest) ++ [First].
```

Odwracanie list

```
-module(list).
-export([rev/1]).
...

rev_tail(L) -> rev_tail(L, []).
rev_tail([], Acc) -> Acc;
rev_tail([First|Rest], Acc) ->
  rev_tail(Rest, [First|Acc]).
```

Actor Model

- lekkie procesy wysyłające wiadomości
- każdy proces ma “skrzynkę odbiorczą”
- każdy proces może wysłać wiadomość

```
echo() ->
  receive
    {Pid, ping} ->
      Pid ! pong,
      echo();
  die -> ok
end.
```

```
$ erl
Eshell V5.9  (abort with ^G)
1> c(echos).
{ok,echos}
2> Pid = spawn(echos, echo, []).
<54.0>
3> Pid ! {self(), ping}.
{<32.0>, ping}
4> flush().
Shell got pong
ok
5>
```

```
echo() ->
  receive
    {Pid, ping} ->
      Pid ! pong,
      echo();
    die -> ok
end.
```

```
echo() ->
  receive
    {Pid, ping} ->
      Pid ! pong,
      echo();
    die -> ok
end.
```

Let it fail™

Let it fail™

- monitor kontroluje stan procesu

Let it fail™

- monitor kontroluje stan procesu
- monitor restartuje procesy

Let it fail™

- monitor kontroluje stan procesu
- monitor restartuje procesy
- błąd w jednym procesie nie zakłóca pracy reszty systemu

Hot Code Loading

Hot Code Loading

- przeładowanie kodu bez restartu systemu

Hot Code Loading

- przeładowanie kodu bez restartu systemu
- jednocześnie mogą działać dwie wersje kodu

Hot Code Loading

- przeładowanie kodu bez restartu systemu
- jednocześnie mogą działać dwie wersje kodu
- ułatwia wprowadzanie poprawek systemu

```
echo() ->
  receive
    {Pid, ping} ->
      Pid ! pong,
      echo();
    die -> ok;
    reload -> ?MODULE:echo()
end.
```

```
$ erl
Eshell V5.9  (abort with ^G)
1> c(echos).
{ok,echos}
2> Pid = spawn(echos, echo, []).
<54.0>
3> Pid ! {self(), "hello"}.
{<32.0>, "hello"}
4> flush().
ok
5>
```

```
echo() ->
  receive
    {Pid, ping} ->
      Pid ! pong,
      echo();
    {Pid, Msg} -> Pid ! Msg,
      echo();
    die -> ok;
    reload -> ?MODULE:echo()
end.
```

```
5> c(echos).
{ok, echos}
6> Pid ! reload.
reload
7> Pid ! {self(), "hello"}.
{<32.0>, "hello"}
8> flush().
Shell got "hello"
Shell got "hello"
ok
9>
```

Kilka słów o OTP

Open Telecom Platform

- zbiór narzędzi i około 50 bibliotek

Open Telecom Platform

- zbiór narzędzi i około 50 bibliotek
- rodzaj standardowej biblioteki

Open Telecom Platform

- zbiór narzędzi i około 50 bibliotek
- rodzaj standardowej biblioteki
- jest dostarczany razem z Erlangiem

Open Telecom Platform

- zbiór narzędzi i około 50 bibliotek
- rodzaj standardowej biblioteki
- jest dostarczany razem z Erlangiem
- przyspiesza tworzenie aplikacji

Co dostajemy z OTP

Co dostajemy z OTP

- odpowiednik `make`

Co dostajemy z OTP

- odpowiednik `make`
- debugger

Co dostajemy z OTP

- odpowiednik `make`
- debugger
- profiler

Co dostajemy z OTP

- odpowiednik `make`
- debugger
- profiler
- silnik bazodanowy

Co dostajemy z OTP

- odpowiednik `make`
- REPL
- debugger
- profiler
- silnik bazodanowy

Co dostajemy z OTP

- odpowiednik `make`
- debugger
- profiler
- silnik bazodanowy
- REPL
- `-behaviours`

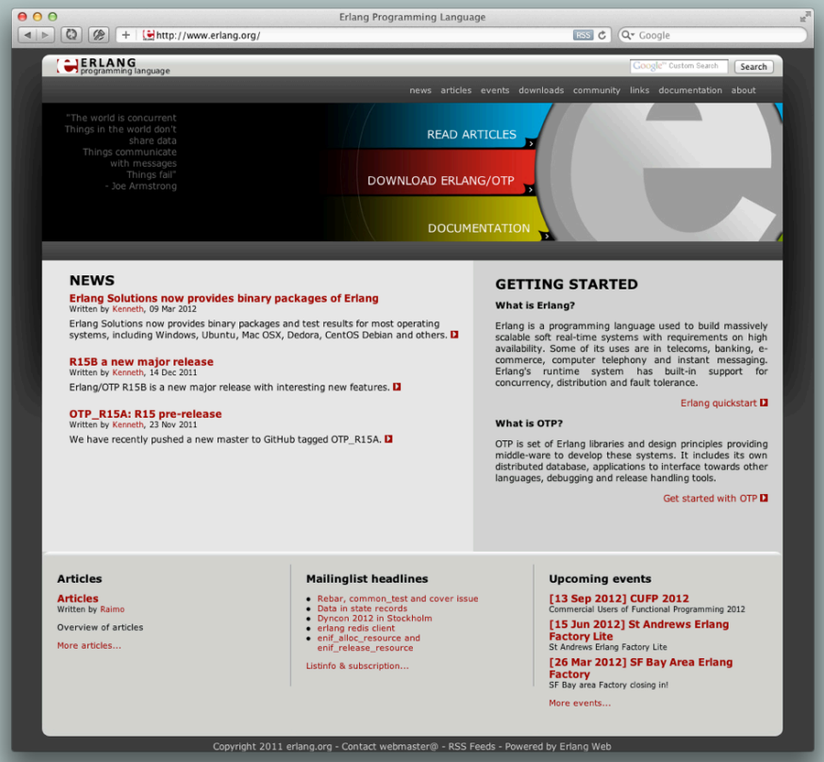
Co dostajemy z OTP

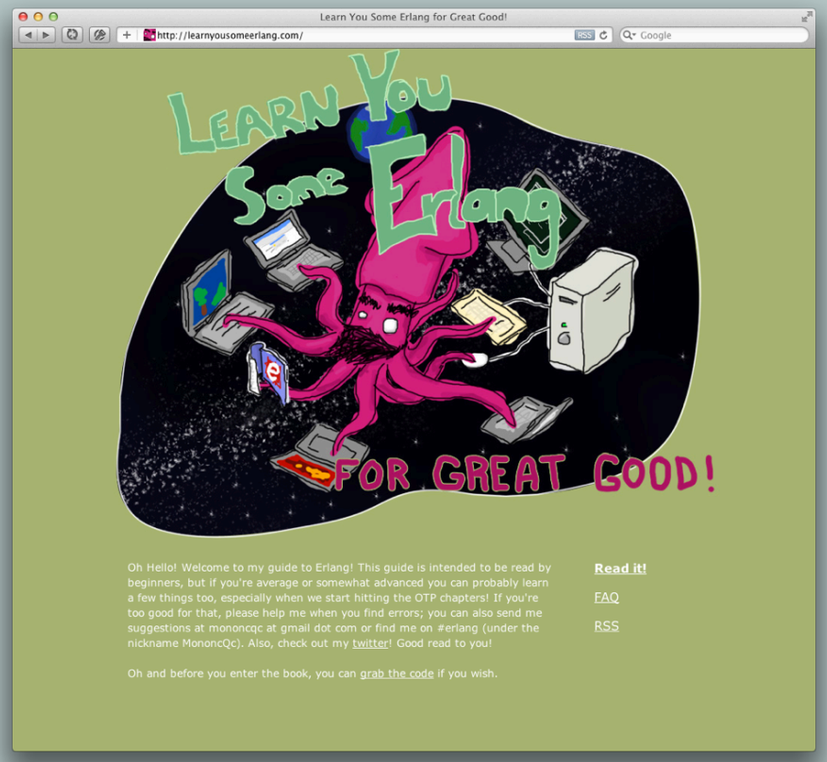
- odpowiednik `make`
- debugger
- profiler
- silnik bazodanowy
- REPL
- `-behaviours`
- interpreter skryptów

Co dostajemy z OTP

- odpowiednik `make`
- debugger
- profiler
- silnik bazodanowy
- REPL
- `-behaviours`
- interpreter skryptów
- ...i wiele innych

Materiały





erlang.org

learnyousomeerlang.com

Dziękuję za uwagę

Pytania?