Erlang

Erlang Top



Erlang Top (etop) jest narzędziem do prezentowania informacji dotyczących procesów Erlanga.

Wynik jego działania podobny jest do informacji prezentowanych przez UNIX-owy <u>top</u>.



Licencja, dostęp

 Copyright Ericsson AB 2002-2013. All Rights Reserved. Licensed under the Apache License, Version 2.0

Etop znajduje się w bibliotece <u>observer</u>



Przykładowy widok

tiger@durin 13:40:32							
0	Memory:	total	1997	binary	33		
197	-	processe	es 0	code	173		
135		atom	1002	ets	95		
Name or Initial Func	Time	Reds	Memory	MsgQ Curren	t Function		
code_server	0	59585	78064	0 gen_se	rver:loop/6		
file_server_2	0	36380	44276	0 gen_se	rver:loop/6		
erl prim loader	0	27962	3740	0 erl pr	im loader:loop		
kernel_sup	0	6998	4676	0 gen_se	rver:loop/6		
net_kernel	62	6018	3136	0 gen_se	rver:loop/6		
init	0	4156	4352	0 init:1	oop/1		
auth	0	1765	1264	0 gen se	rver:loop/6		
inet tcp dist:accept	0	660	1416	0 prim i	prim inet:accept0/2		
application controll	0	569	6756	0 gen se	0 gen server:loop/6		
net_kernel:do_spawn_	0	553	5840	0 dbg:do	_relay_1/1		
	197 135 Name or Initial Func code_server file_server_2 erl_prim_loader kernel_sup net_kernel init auth inet_tcp_dist:accept application_controll	197 135 Name or Initial Func Time code_server	197 processe 135 atom Name or Initial Func Time Reds code_server 0 59585 file_server_2 0 36380 erl_prim_loader 0 27962 kernel_sup 0 6998 net_kernel 62 6018 init 0 4156 auth 0 1765 inet_tcp_dist:accept 0 660 application_controll 0 569	197 processes 0 atom 1002 Name or Initial Func Time Reds Memory code_server 0 59585 78064 file_server_2 0 36380 44276 erl_prim_loader 0 27962 3740 kernel_sup 0 6998 4676 net_kernel 62 6018 3136 init 0 4156 4352 auth 0 1765 1264 inet_tcp_dist:accept 0 660 1416 application_controll 0 569 6756	197		



tiger@durin - nazwa węzła, którego procesy obserwujemy

Load:

- cpu procent czasu, kiedy węzeł był aktywny,
- procs liczba procesów danego węzła,
- runq liczba procesów, które są gotowe do uruchomienia.

tiger@durin Load: cpu procs runq	0 197 135	Memory:	total processe atom	1997 es 0 1002	binary code ets	13:40:32 33 173 95
Pid	Name or Initial Func	Time	Reds	Memory	MsgQ Currer	nt Function
<127.23.0> <127.21.0> <127.2.0> <127.9.0> <127.17.0> <127.0.0> <127.16.0> <127.18.0> <127.5.0> <127.137.0>	code_server file_server_2 erl_prim_loader kernel_sup net_kernel init auth inet_tcp_dist:accept application_controll net_kernel:do_spawn_	0 0 0 62 0 0 0	59585 36380 27962 6998 6018 4156 1765 660 569 553	78064 44276 3740 4676 3136 4352 1264 1416 6756 5840	0 gen_se 0 erl_pr 0 gen_se 0 gen_se 0 init:1 0 gen_se 0 prim_i 0 gen_se	erver:loop/6 erver:loop/6 eim_loader:loop erver:loop/6 erver:loop/6 loop/1 erver:loop/6 inet:accept0/2 erver:loop/6



13:40:32 – aktualny czas [hh:mm:ss]

Memory – pamięć zaalokowana przez węzeł w kilobajtach

tiger@durin Load: cpu procs runq	0 197 135	Memory:	total processe atom	1997 es 0 1002	binary code ets	13:40:32 33 173 95
Pid	Name or Initial Func	Time	Reds	Memory	MsgQ Current	Function
<127.23.0> <127.21.0> <127.2.0> <127.9.0> <127.17.0> <127.16.0> <127.16.0> <127.18.0> <127.5.0> <127.137.0>	code_server file_server_2 erl_prim_loader kernel_sup net_kernel init auth inet_tcp_dist:accept application_controll net_kernel:do_spawn_	0 0 0 62 0 0 0	59585 36380 27962 6998 6018 4156 1765 660 569 553	78064 44276 3740 4676 3136 4352 1264 1416 6756 5840	0 gen_ser 0 erl_pr: 0 gen_ser 0 gen_ser 0 init:10 0 gen_ser 0 prim_ir 0 gen_ser	rver:loop/6 rver:loop/6 im_loader:loop rver:loop/6 rver:loop/6 rver:loop/6 rver:loop/6 rver:loop/6 rver:loop/6 rver:loop/6 rver:loop/6 rver:loop/6



Dla każdego procesu wyświetlane są następujące informacje:

- *Pid* identyfikator procesu,
- Name... nazwa procesu,
- Time czas działania procesu,
- Reds

 liczba redukcji (każda operacja w systemie np. wywołanie funkcji w pętli, BIF-y itp.) wykonanych w procesie,

tiger@durin Load: cpu procs runq	0 197 135	Memory:	total processe atom	1997 es 0 1002	binary code ets	13:40:32 33 173 95
Pid	Name or Initial Func	Time	Reds	Memory	MsgQ Current	: Function
<127.23.0>	code_server	0	59585	78064		ver:loop/6
<127.21.0>	Ille_server_2	U	36380	44276	u gen_ser	ver:100p/6
<127.2.0>	erl_prim_loader	0	27962	3740	0 erl_pri	m_loader:loop
<127.9.0>	kernel sup	0	6998	4676	0 gen ser	ver:loop/6
<127.17.0>	net kernel	62	6018	3136	0 gen ser	ver:loop/6
<127.0.0>	init	0	4156	4352	0 init:lo	oop/1
<127.16.0>	auth	0	1765	1264	0 gen ser	ver:loop/6
<127.18.0>	inet tcp dist:accept	0	660	1416		net:accept0/2
<127.5.0>	application controll	0	569	6756		ver:loop/6
<127.137.0>	net_kernel:do_spawn_	0	553	5840		relay_1/1



- Memory rozmiar procesu w bajtach, otrzymywany przez wywołanie process_info(pid, memory),
- MsgQ (message queue) długość kolejki wiadomości procesu,
- Current Function obecnie wywołana funkcja.

tiger@durin Load: cpu procs runq	0 197 135	Memory:	total processe atom	1997 es 0 1002	binary code ets	13:40:32 33 173 95
Pid	Name or Initial Func	Time	Reds	Memory	MsgQ Current	: Function
<127.23.0>	code_server	0	59585	78064	0 gen_ser	ver:loop/6
<127.21.0>	Ille_server_Z	U	36380	44276	u gen_ser	ver:100p/6
<127.2.0>	erl_prim_loader	0	27962	3740	0 erl_pri	m_loader:loop
<127.9.0>	kernel sup	0	6998	4676	0 gen ser	ver:loop/6
<127.17.0>	net kernel	62	6018	3136	0 gen ser	ver:loop/6
<127.0.0>	init	0	4156	4352	0 init:lo	op/1
<127.16.0>	auth	0	1765	1264	0 gen ser	ver:loop/6
<127.18.0>	inet tcp dist:accept	0	660	1416		et:accept0/2
<127.5.0>	application controll	0	569	6756		ver:loop/6
<127.137.0>	net_kernel:do_spawn_	0	553	5840		relay_1/1



Narzędzie etop powinniśmy uruchamiać za pomocą odpowiedniego skryptu:

Windows:



UNIX:



```
#!/bin/sh

NAME="etop"
erl -sname $NAME -hidden -s etop -s erlang halt -output text $0
```



Węzeł, który chcemy monitorować należy podać jako parametr (atom) w konsoli w formacie:

-node nazwa@host

Przykładowy skrypt może wyglądać następująco:



Dostępne są ponadto opcjonalne parametry uruchamiania:

-setcookie cookieName

Ciasteczko dla węzła *etop*. Musi być takie samo jak dla węzła monitorowanego. Argument: atom.

· -lines 10

Ilość linii (procesów) do wyświetlenia. Argument: integer.

-interval 10

Odstęp w sekundach między kolejnymi wyświetleniami.

Argument: integer.



• -accumulate true/false

Określa jak prezentowane są nagłówki Time oraz Reds. Domyślnie false.

true → od ostatniego wyświetlenia

false → od początku

Argument: atom (true lub false).

-sort memory

Ustawienia sortowania. Domyślnie runtime (jeżeli tracing=off to reductions). Dostępne argumenty (atomy): runtime | reductions | memory | msg_q

· -tracing on/off

Etop używa erlangowskiej funkcji śledzenia, dlatego nie ma innej możliwości monitorowania węzła, dopóki funkcja *tracing* jest uruchomiona. Dostępne argumenty: on lub off.



Funkcje

Możemy użyć także następujących funkcji z modułu etop:

- start() → ok
 Uruchomienie narzędzia etop (preferuje się jednak użycie skryptu).
- start([Options]) → ok
 Uruchomienie z dodatkowymi parametrami. Opcje podajemy jako krotki 2-atomowe.
 Możliwe jest zastosowanie takich parametrów, jak te startowe wpisywane w konsolę (wyjątek: cookie jako string) oraz dodatkowo <u>ustawienie portu</u>: {port, integer}.
- config(Key, Value) → Result
 Zmiana parametrów w czasie działania narzędzia.
- dump(File) → Result
 Zapis wyświetlanych danych do pliku podanego jako argument (string).
- stop() → stop
 Zatrzymanie narzędzia etop.