Learn You Some Erlang for Great Good!

A Beginner's Guide

by Fred Hebert

errata updated to print 8

Page	Error	Correction	Print corrected
38	Note that the *.beam* file generated will **no longer be portable across platforms**.	Note that the *.beam* file generated will **contain both native and non-native code, and the native part will not be portable across platforms**.	Print 3
62	The factorial of a number n is the product of the sequence $1 \times 2 \times 3 \times ... \times n$, or alternatively $n \times \mathbf{n - 1 \times n - 2} \times ... \times 1$.	The factorial of a number n is the product of the sequence $1 \times 2 \times 3 \times ... \times n$, or alternatively $n \times \mathbf{(n - 1) \times (n - 2)} \times ... \times 1$.	Print 3
128	In naive mode, the functions are `gb_trees:enter/2`, `gb_trees:lookup/2`, and `gb_trees:delete_any/2`.	In naive mode, the functions are `gb_trees:enter/3`, `gb_trees:lookup/2`, and `gb_trees:delete_any/2`.	Print 3
138	URL replacement	Some studies proved that the main sources of downtime in large-scale software systems are intermittent or transient bugs (see *https://dslab.epfl.ch/pubs/crashonly.pdf*).	Print 3
153	1. A message to **store** food is sent from you (the shell) to the fridge process. . . . 3. The fridge **stores** the item and sends **ok** to your process.	1. A message to **take** food is sent from you (the shell) to the fridge process. . . . 3. The fridge **removes** the item and sends **it** to your process.	Print 3
154	1. A message to **store** food is sent from you (the shell) to an unknown process.	1. A message to **take** food is sent from you (the shell) to an unknown process.	Print 3
188	`error:function_clause -> %% not in {{D,M,Y},{H,Min,S}} format` `  false`	`error:function_clause -> %% not in {{Y,M,D},{H,Min,S}} format` `  false`	Print 3
224	The `sitting/2` function can then return the tuples {next_state, NextStateName, NewStateData}, `{next_state, NextStateName, NewStateData, Timeout}`, `{next_state, NextStateName, hibernate}`, and `{stop, Reason, NewStateData}`.	The `sitting/2` function can then return the tuples {next_state, NextStateName, NewStateData}, `{next_state, NextStateName, NewStateData, Timeout}`, `{next_state, NextStateName, NextStateData, hibernate}`, and `{stop, Reason, NewStateData}`.	Print 3
269	`StartFunc` is a tuple that specifies how to start the **supervisor**.	`StartFunc` is a tuple that specifies how to start the **child**.	Print 3
269	So instead of doing `supervisor:start_child(Sup, Spec)`, which would call `erlang:apply(M,F,A)`, we now have `supervisor:start_child(Sup, Args)`, which calls `erlang:apply(M,F,Args++A)`.	So instead of doing `supervisor:start_child(Sup, Spec)`, which would call `erlang:apply(M,F,A)`, we now have `supervisor:start_child(Sup, Args)`, which calls `erlang:apply(M,F,A++Args)`.	Print 3

Page	Error	Correction	Print corrected

291	`-define(SPEC(MFA),`
	`  {worker_sup,`
	`  {ppool_worker_sup, start_link, [MFA]},`
	`  permanent,`
		`-define(SPEC(MFA),`
		`  {worker_sup,`
		`  {ppool_worker_sup, start_link, [MFA]},`
		`  temporary,`
			Print 3

291	`init({Limit, MFA, Sup}) ->`
	`  {ok, Pid} = supervisor:start_child(Sup, ?SPEC(MFA)),`
	`  {ok, #state{limit=Limit, refs=gb_sets:empty()}}.`
		`init({Limit, MFA, Sup}) ->`
		`  {ok, Pid} = supervisor:start_child(Sup, ?SPEC(MFA)),`
		`  link(Pid),`
		`  {ok, #state{limit=Limit, refs=gb_sets:empty()}}.`
			Print 3

292	`handle_info({start_worker_supervisor, Sup, MFA}, S = #state{}) ->`
	`  {ok, Pid} = supervisor:start_child(Sup, ?SPEC(MFA)),`
	`  {noreply, S#state{sup=Pid}};`
		`handle_info({start_worker_supervisor, Sup, MFA}, S = #state{}) ->`
		`  {ok, Pid} = supervisor:start_child(Sup, ?SPEC(MFA)),`
		`  link(Pid),`
		`  {noreply, S#state{sup=Pid}};`
			Print 3

307	This tells OTP that when starting your application, it should call `CallbackMod:start(normal, Args)`. **It will also call `CallbackMod:stop(Args)` when stopping it.**	This tells OTP that when starting your application, it should call `CallbackMod:start(normal, Args)`. **This function's return value will be used when OTP will call `CallbackMod:stop(StartReturn)` when stopping your application.**	Print 3
341	If you're using pure Erlang code **without native compiling with HiPE (a native compiler for Erlang code, which gives somewhat faster code, especially for CPU-bound applications), then that code will be portable.**	If you're using pure Erlang code**, then that code will be portable.**	Print 5
351, 367	`{app, stdlib, [{mod_cond, derived}, {incl_cond, include}]}`	`{app, stdlib, [{incl_cond, include}]}`	Print 3
383	Note that closing an accept socket will close that socket alone, and closing a listen socket will close **all of the related accept sockets.**	Note that closing an accept socket will close that socket alone, and closing a listen socket will close **none of the related and established accept sockets, but will interrupt currently running calls to accept new ones.**	Print 3
395	`handle_info({tcp_closed, _Socket, _}, S) ->`	`handle_info({tcp_closed, _Socket}, S) ->`	Print 3
402	It's called that because, secretly, the underlying implementation of `?_assert(A == B)` is `fun() -> ?assert(A,B) end;` that is to say, it's a function that generates a test.	It's called that because, secretly, the underlying implementation of `?_assert(A == B)` is `fun() -> ?assert(A==B) end;` that is to say, it's a function that generates a test.	Print 3

Page	Error	Correction	Print corrected
406	`some_test2_() ->` `  {foreach`	`some2_test_() ->` `  {foreach`	Print 3
407	`some2_tricky_test_()`	`some_tricky_test2_()`	Print 3
435	`?MODULE = ets:new(regis, [set, named_table, protected])`	`?MODULE = ets:new(?MODULE, [set, named_table, protected])`	Print 3
439	Note that we use `regis (?MODULE)` as the table name here . . .	Note that we use `regis_server (?MODULE)` as the table name here . . .	Print 3
459, 463, 465, 534, 536, 538	`net_kernel:connect`	`net_kernel:connect_node`	Print 5
465	Aha! The node didn't connect to **`ketchup`** . . .	Aha! The node didn't connect to **`salad`** . . .	Print 3
466	The **`ketchup`** node will never see any connection to `olives` . . .	The **`salad`** node will never see any connection to `olives` . . .	Print 3
466	. . . set a range of **15** ports to be used for Erlang nodes.	. . . set a range of **16** ports to be used for Erlang nodes.	Print 3
467	By default, the heartbeat delay ~~(also called tick time)~~ is set to 15 seconds, or 15,000 milliseconds.	By default, the heartbeat delay is set to 15 seconds, or 15,000 milliseconds. **After 4 failed heartbeats, a remote node is considered dead. The heartbeat delay multiplied by 4 is called the tick time.**	Print 3
506	`{logdir, [all_nodes, master], "./logs/"}.`	`{logdir, all_nodes, "./logs/"}.` `{logdir, master, "./logs/"}.`	Print 3
506	To truly include all nodes, **`[all_nodes, master]` is** required.	To truly include all nodes, **both `all_nodes` and `master` are** required.	Print 3
545	`foo(X) when is_integer(X) -> X + 1.`	`foo(X) when is_integer(X) -> X + 1;`	Print 3