Errata for *Learn You Some Erlang for Great Good!* (updated to 8th printing)

Page 38: The note that reads:
"Note that the *.beam* file generated will no longer be portable across platforms."
should now read:
"Note that the *.beam* file generated will contain both native and non-native code, and the native part will not be portable across platforms."

Page 62: The equation that reads:
$n \times n - 1 \times n - 2 \times ... \times 1$
should now read:
$n \times (n - 1) \times (n - 2) \times ... \times 1$
to respect operator precedence.

Page 128: `gb_trees:enter/2` should now read `gb_trees:enter/3`

Page 138: The URL *http://dslab.epfl.ch/pubs/crashonly/* should now read *http://dslab.epfl.ch/pubs/crashonly.pdf*

Page 153: The numbered list items that read:
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.
should now read:
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.

Page 154: The numbered list item that reads:
1. A message to store food is sent from you (the shell) to an unknown process.
should now read:

1. A message to take food is sent from you (the shell) to an unknown process.

Page 188: In the block of code at the bottom of the page, the error which reads:

```
error:function_clause -> %% not in {{D,M,Y},{H,Min,S}} format
      false
```

should now read:

```
error:function_clause -> %% not in {{Y,M,D},{H,Min,S}} format
      false
```

Page 224: The value mentioned in the text that reads:

```
{next_state, NextStateName, hibernate}
```

should now read:

```
{next_state, NextStateName, NextStateData, hibernate}
```

Page 269: In the StartFunc section, the first line that reads:

". . . how to start the supervisor."

should now read:

". . . how to start the child."

Page 279: In the last sentence of the second paragraph, the code which reads:

```
erlang:apply(M,F,Args++A)
```

should now read:

```
erlang:apply(M,F,A++Args)
```

Page 291: The block of code in the center of the page which reads:

```
%% The friendly supervisor is started dynamically!
-define(SPEC(MFA),
    {worker_sup,
    {ppool_worker_sup, start_link, [MFA]},
    permanent,
    10000,
    supervisor,
    [ppool_worker_sup]}).
```

should now read:

```
%% The friendly supervisor is started dynamically!
-define(SPEC(MFA),
    {worker_sup,
    {ppool_worker_sup, start_link, [MFA]},
    temporary,
    10000,
    supervisor,
    [ppool_worker_sup]}).
```

and the block of code at the bottom of the page which reads:

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

should now read:

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

Page 292:

The block of code at the bottom of the page which reads:

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

should now read:

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

Page 307: The text reads:

"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."

should now read:

“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.”

Page 341: The text which reads:
“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.”
should now read:
“If you’re using pure Erlang code, then that code will be portable.”

Page 351: In the code at the top of the page, the line which reads:
```
{app, stdlib, [{mod_cond, derived}, {incl_cond, include}]},
```
should now read:
```
{app, stdlib, [{incl_cond, include}]},
```

Page 367:
In the code block in the center of the page, the line which reads:
```
{app, stdlib, [{mod_cond, derived}, {incl_cond, include}]},
```
should now read:
```
{app, stdlib, [{incl_cond, include}]},
```

and the text which reads:
“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.”
should now read:
“If you’re using pure Erlang code, then that code will be portable.”

Page 383: The line which reads:
“Note that closing an accept socket will close that socket alone, and closing a listen socket will close all of the related accept sockets.”

should now read:

"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."

Page 395: In the last block of code at the bottom of the page, the line that reads:

```
handle_info({tcp_closed, _Socket, _}, S) ->
```

should now read:

```
handle_info({tcp_closed, _Socket}, S) ->
```

Page 402: In the third paragraph under "Test Generators," the sentence that reads:

"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."

should now read:

"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."

Page 406: The first two lines of the `some_test_/0` function that read:

```
some_test2_() ->
      {foreach
```

should now read:

```
some2_test_() ->
      {foreach;
```

Page 407: The first line of the first code block that reads:

```
some_tricky_test2_()
```

should now read:

```
some_tricky2_test_()
```

Page 435: The line of code in the section titled "Implementation Details" which reads:

```
      ?MODULE = ets:new(regis, [set, named_table, protected]),
```

should now read:

```
      ?MODULE = ets:new(?MODULE, [set, named_table, protected]),
```

Page 439: In the paragraph following the first code block, the line:
"Note that we use `regis (?MODULE)` as the table name here . . ."
should now read:
"Note that we use `regis_server (?MODULE)` as the table name here . . ."

Pages 459, 463, 465, 534, 536, and 538:
All instances of the function `net_kernel:connect` should now read
`net_kernel:connect_node`

Page 465: In the last paragraph, `ketchup` should now read `salad`.

Page 466: In the first paragraph, `ketchup` should now read `salad`.

and in the sixth paragraph, "15 ports" should now read "16 ports."

Page 467: The text reads:
"By default, the heartbeat delay (also called tick time) is set to 15 seconds, or 15,000 milliseconds."
should now read:
"By default, the heartbeat delay is set to 15 seconds, or 15,000 milliseconds. After 4 failed heart-beats, a remote node is considered dead. The heartbeat delay multiplied by 4 is called the tick time."

Page 506: In the code block in the middle of the page:
```
{logdir, [all_nodes, master], "./logs/"}.
```
should now read:
```
{logdir, all_nodes, "./logs/"}.
{logdir, master, "./logs/"}.
```

and the sentence in the second to last paragraph of the page which reads:
"To truly include all nodes, `[all_nodes, master]` is required"

should now read:

“To truly include all nodes, both `all_nodes` and `master` are required.”

Page 545: The first line of code in the middle of the page which reads:

```
foo(X) when is_integer(X) -> X + 1.
```

should now read:

```
foo(X) when is_integer(X) -> X + 1;
```