Concurrent Programming in Erlang

So, we've come to the end of the first week. This week we've looked at:

- The fundamentals of concurrency in Erlang - processes that "share nothing" and that communicate by passing messages.
- The mechanisms underlying Erlang concurrency - the `receive` construct, mailboxes, Pids and named processes.
- The frequency server case study, providing a larger-scale example of Erlang concurrency.

We will also provide more specific feedback, based on your questions and contributions across this week's comments, and addressing any common questions or themes that emerge during the week. We'll look to provide a link here to this feedback on Monday 10th April, by 12 noon (UK time).

Next week we'll look at ways in which things can go wrong in communicating systems, and the Erlang approach to dealing with those.