Elixir's Set-Theoretic Type System

Robert Ellen

2024/11/12

Summary

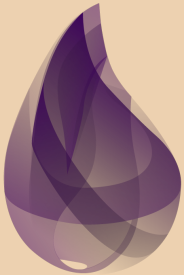
Elixir

Elixir

What is the Erlang Ecosystem?

A group of languages, libraries, frameworks, and applications that are implemented on top of the Erlang virtual machine, the BEAM.

Languages include:



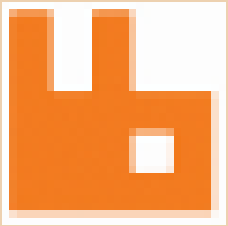
elixir

...plus dozens more

What is the Erlang Ecosystem?

Libraries and frameworks include:

OTP



What is the Erlang Ecosystem?

Built around a shared value in:

- massive concurrency
- fault-tolerance
- simplicity
- acknowledging the errors will occur so lets deal with them
  - “Let it crash”–have a plan to restart sub-systems when they crash

Brief history of Erlang

covered in my 2013 talk, but tonight...

Brief history of Erlang

Brief history of Erlang

- developed in the mid 1980s at Ericsson
- to run on next generation telephone switches
  - concurrent, fault-tolerant, distributed, soft real-time
  - reports of 1200k LOC and “nine nines” of uptime on the AXD301 switch
  - reports of market penetration of > 50% in mobile telephony switches
- solved web-scale in the '80s

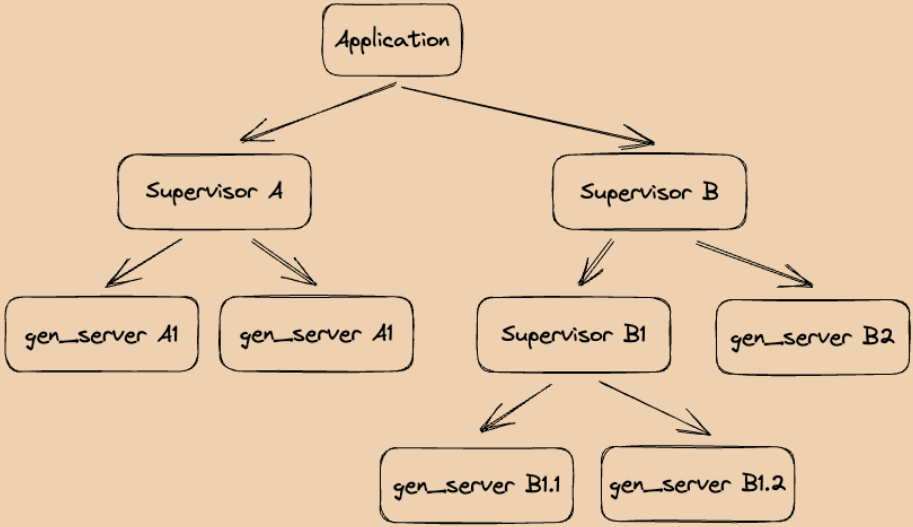
The BEAM

- Erlang's virtual machine / runtime system
- lightweight processes, SMP
- multi-generational, per-process garbage collector
- asynchronous, location-transparent, message-passing IPC
- hot-code-loading
- powerful REPL and introspection tools
- new JIT compiler (BEAM bytecode to machine code)

The OTP libraries

- Erlang's ”standard library”
- dozens of modules providing various typical stdlib stuff
- core of which are for supervision trees
  - `gen_server` - actors / workers
  - supervisors - handling starting/stopping/restarting `gen_servers`

supervision trees



Elixir

There will be some comparing and contrasting of Gleam with Elixir...

Brief history of Elixir

> created by José Valim starting in 2012
> inspired by Erlang, Ruby, and, to a lesser extent, Lisp
> like Erlang, the language is quite stable

elixir

Features of Elixir

- Ruby-like syntax while retaining most of Erlang's semantics
  - ..., immutable data, HoF, side-effects anywhere, dynamically-typed, ...
- interoperate with Erlang
- hygienic macros
- highly ergonomic build tool: `mix`
- modern package manager: `hex`
- excellent unit test tool: exunit
- opinionated formatter
- drops strict SSA in favour of rebinding

Typing BEAM languages

Marlow & Wadler - 1997

- “We can stop waiting for functional languages to be used in practice–that day is here!”
- threw away Hindley-Milner: $U = V$ – this would not work with existing Erlang codebases
- proposed strictly more general sub-typing instead: $U \subseteq V$

eqWAlizer

Developed by Meta for WhatsApp

Typed BEAM languages with alternate semantics

- Hamler - purescript for the BEAM
- Caramel - ML for the BEAM
- Gleam - HM-based type system - see my May 2024 talk
- ...

Static analysis tools

- dialyzer - Linhahl & Sagonas, 2006
- gradualizer

References

References

Simon Marlow and Philip Wadler. A practical subtyping system for erlang. In Proceedings of the Second ACM SIGPLAN International Conference on Functional Programming, ICFP '97, page 136–149, New York, NY, USA, 1997. Association for Computing Machinery. doi:10.1145/258948.258962.

Tobias Lindahl and Konstantinos Sagonas. Practical type inference based on success typings. In ACM-SIGPLAN International Conference on Principles and Practice of Declarative Programming, 2006. doi:10.1145/1140335.1140356.

Thank you

I'll post slides soon.