Vladimir Logachev

Fullstack developer, FP enthusiast.
Remote (Novosibirsk, Russia)

logachev.dev@ya.ru

Site: https://logachev.dev
GitHub: https://github.com/vladimirlogachev
Telegram: https://t.me/vladimirlogachev
Twitter: https://twitter.com/logachev_dev
LinkedIn: https://www.linkedin.com/in/vladimirlogachev

About me

I prefer functional languages that implement strict static typing. I use Haskell, Elm, and Scala, but am also ready to tackle any other typed functional language.

I associate the success in my career with FP, so I devote a lot of time and attention not only to code but also to people. My mission as a developer is to make functional programming deliver value to both companies and individual specialists.

I'm a big fan of meetups and reading groups, which I run at my workplaces from time to time, and also I consider pair programming and pair testing to be an effective practice.

In programming, I prefer not to rely on intuition (which, I believe, is usually based on previous experiences and tends to fail in unprecedented situations), but instead, read books well in advance.

Skills and usage experience

Haskell

Concepts: Monads, applicatives, monad transformers
Libraries: mu-hakell, postgres-typed, aeson, parsec, transformers, mtl, wai, servant
Language extensions: TypeApplications, TypeOperators, PartialTypeSignatures, DeriveFunctor, StandaloneDeriving, OverloadedStrings

Scala

FP: cats-core, cats-effect, fs2, scala-parser-combinators
Testing: scalatest, scalacheck, specs2
Other libraries: scodec, akka, akka-http, akka-stream, scala-parser-combinators

Elm

Concepts: Tasks, Ports, JSON encoding/decoding, Browser API Interop (Websockets)
Libraries: elm-css, elm-graphql, elm-ordering, elm-units, elm-dropbox, elm-crypto-string

Other

Databases: PostgreSQL, Redis, Clickhouse, MongoDB
Infrastructure and tooling: Docker, Dhall, GitHub Actions
APIs: GraphQL

Showcase projects and assessments

servant-to-elm example

https://github.com/VladimirLogachev/servant-to-elm-example

An example full-stack web application, built in a typesafe functional way. What's cool there is that servant-to-elm does the job of generating types and decoders/encoders from Haskell types and Servant definition to Elm, which not only catches regressions in the compile-time but also provides ready (and highly configurable) Elm functions to fetch necessary data from the server.
Elm, Haskell, Servant, SQLite

Transitive Closure (assessment)

https://github.com/VladimirLogachev/transitive_closure

A function that accepts list of object ids and returns those objects and all objects which they refer to (directly or indirectly) from some Repository with monadic interface. The code is pretty abstract, but still well-tested (including tests for cases like very large referencing graphs and cyclic references).
Scala, Cats, ScalaTest

Web crawler microservice (assessment)

https://github.com/VladimirLogachev/crawler

A microservice that accepts a list of page urls and returns a list of page titles. It takes into account situations like bad urls, duplicate urls, redirects, concurrency and backpressure.
Scala, Akka HTTP

Notable contributions

higherkindness/mu-graphql-example-elm

https://github.com/higherkindness/mu-graphql-example-elm

An example of how to implement both frontend and backend in a schema-first, typesafe, and functional way (for the mu-haskell library, demonstrating its GraphQL capabilities).
I rebuilt its Elm frontend and made minor changes to Haskell backend (and also discovered a couple of bugs).
Elm, Haskell, GraphQL

Russian translation of the Mostly Adequate Guide to Functional Programming in JavaScript

https://github.com/MostlyAdequate/mostly-adequate-guide-ru

The book introduces the reader to the functional programming paradigm and describes a functional approach to developing JavaScript applications. The translation was initiated by Maxim Filippov and stopped at 60%. Then me and Sakayama joined the translation, refactored every chapter translated before us and then finished the translation.
JavaScript

FP Specialty

https://fpspecialty.github.io

I lead a functional programming reading group for English speaking users of all functional programming skills.
Reading group

Education and courses

Mastering Haskell Programming

https://www.udemy.com/certificate/UC-DRMAMOQ5
Packt, 2019

Functional Programming in Haskell, part 2 (certificate with honors)

https://stepik.org/cert/207739
Computer Science Center, 2019

Functional Programming in Haskell, part 1 (certificate with honors)

https://stepik.org/cert/196007
Computer Science Center, 2019

Computer Science Summer School, Theory of Programming Languages

Novosibirsk State University, 2019

Maintenance of computer equipment and computer networks

Novosibirsk Aviation Technical College, 2004 — 2008

Experience

ElectroNeek Robotics, Inc, frontend developer 01/2021 — present

https://electroneek.com
Backend: TypeScript, MongoDB, Nest.js, Jest; Infrastructure: Gitlab CI

Pamir, frontend developer 05/2020 — 12/2020

Developed a web application, which utilizes server-side rendering and covered it with unit tests. Packaged everything in Docker and set up CI. I also mentored the second frontend developer who joined the team later.
Frontend: TypeScript, React, Next.js, GraphQL, Apollo, FP-TS, Emotion, Jest; Infrastructure: Nginx, Docker, GitHub Actions

Eldis, software engineer 10/2019 — 12/2019

https://eldis.ru
I developed a declarative decoder for the internal binary document format, covered it with tests, including property-based testing.
Scala, scodec, cats, fs2, decline, specs2, scalacheck

Neolab-Nsk, fullstack developer 01/2019 — 09/2019

I implemented new functionality in existing web applications, fixed defects and developed new applications, and microservices, covered them with unit tests and integration tests.
Frontend: TypeScript, React, Redux, Saga, RxJS, FP-TS; Backend: TypeScript, Node, Redux, Saga, RxJS, Redis, Lua, Mongo, PostgreSQL, Clickhouse, Docker

SocialSweet Inc, frontend developer 08/2018 — 01/2019

https://sweet.io
Sweet's product is a loyalty platform, social network and online store. I performed tasks related to business logic at the front end and was engaged in covering the existing code with unit tests and tuning them, thanks to which the tests were launched using CI pipeline, and the defects associated with an unsuccessful merging of Git branches in a huge codebase really began to be prevented.
TypeScript, Angular, RxJS

Allmax, frontend developer 11/2017 — 08/2018

https://savl.com/

I worked in the Savl project — this is a mobile application, wallet with support for 6 cryptocurrencies. I was responsible for the data layer in the mobile application. I applied everything that I learned from books about functional programming and software design, and also completely covered the business logic with tests, as a result of which the application became fault-tolerant and modular, that is, it stopped crashing due to exceptions or unexpected behavior of external services, and allowed to enable and disable support for individual cryptocurrencies at any time. Also inside the company, I made several presentations on functional programming.

JavaScript, Flow, React Native, Redux, Saga, Ramda, Sanctuary, Socket.io