SANDY MAGUIRE

+1 250 986 0250 | sandy@sandymaguire.me | isovector

Programming Experience

SUMMARY OF SKILLS

↪ Haskell (expert) ↪ Agda, Scala (fluent) ↪ C++, C#, JavaScript, Lua, PHP, Python (working proficiency)

Manifold Valley » Lead Compiler Engineer (Haskell)

April 2023 → ongoing

WORK EXPERIENCE

↪ Prototyped and implemented a new compiler for a functional choreographic programming language.

↪ Optimized the existing project's compile times (in GHC) from 10 minutes down to 30 seconds.

↪ Replaced the existing language's custom type system with algebraic data types; increased developer productivity by over 200%.

↪ Dramatically improved runtime performance by implementing normalization by evaluation.

↪ Implemented (and insisted on) property- and golden-testing frameworks for verifying compiler output.

Wire » Consultant (Haskell)

October 2021 → April 2023

↪ Built a GHC plugin to track and reify federated service calls at the type-level.

↪ Designed a property-testing framework for verifying the correctness of higher-order algebraic effects.

Cofree Press » Author of Software Textbooks

March 2018 → November 2023

https://leanpub.com/u/sandy-maguire

↪ Wrote three books on advanced programming techniques and high-quality software engineering.

↪ Algebra-Driven Design is now the basis of a course taught at OST Zurich.

Formation/Takt » Senior Software Engineer (Haskell)

September 2016 → January 2018

↪ Increased new feature cadence by 30x after becoming lead of a four-person engineering team.

↪ Directed a team of three to implement a high-throughput, low-latency brokered streaming library.

Google » Software Engineer III (C++)

September 2015 → September 2016

↪ Led the architectural design of a user-defined permission model for the cloud.

↪ Improved compile times by 96% and test coverage by 65% for a service-critical internal compiler.

Meta/Facebook » Software Engineer Intern (C++)

January → April 2014

↪ Increased revenue by 0.5% after analyzing the advertising platform's spending behaviors.

↪ Improved site-wide response time by 0.4% by parallelizing the backend graph ranker.

Cornelis

2022 → 2024

NOTABLE OPEN SOURCE

github.com/isovector/cornelis

↪ Integrated Neovim tightly with the Agda compiler, allowing for interactive proof assistance.

ImplicitCAD

2020 → 2021

github.com/Haskell-Things/ImplicitCAD

↪ Improved performance of single-core mesh rendering by ~2x.

↪ Reduced code duplication by 50% by reorganizing types to be shared between 2D and 3D.

Wingman for Haskell

2020 → 2023

github.com/haskell/haskell-language-server

↪ Developed an interactive tactic engine for Haskell, capable of robust, type-aware code synthesis.

↪ Provided in-editor support for automatic pattern splitting.

Algebra Checkers

2020

github.com/isovector/algebra-checkers

↪ Wrote a model checker for verifying the consistency of combinator libraries' algebras.

Polysemy

2019 → 2023

github.com/polysemy-research/polysemy

↪ Discovered a convenient encoding of an effect system based on higher-order free monads via simultaneous co-Yoneda and codensity transformations.

↪ Implemented a GHC plugin to support ad-hoc functional dependencies when working with Polysemy; dramatically improving the developer experience.

PUBLICATIONS

Certainty by Construction
November 2023

leanpub.com/certainty-by-construction

↪ An exploration of topics from mathematics and computer science, entirely in literate Agda.

Algebra-Driven Design
September 2020

leanpub.com/algebra-driven-design

↪ A series of worked examples on designing and efficiently implementing combinator libraries.

Thinking with Types
October 2018

thinkingwithtypes.com

↪ A how-to manual on using (and not misusing) Haskell's more advanced type-level features.

How These Things Work
November 2017

reasonablypolymorphic.com/book/preface.html

↪ A technical and philosophical journey into how computers work, starting from first principles.

FORMAL EDUCATION

Master of Computer Science
2023 → 2024 (voluntarily withdrawn)

Software Practices Lab, University of British Columbia, Vancouver, BC

Relevant Courses

↪ Dependent Types – implemented a small dependently typed language

↪ Program Verification – implemented a SAT solver and program verifier for a C-like language

Bachelor of Software Engineering
2010 → 2015

University of Waterloo, Waterloo, ON

Relevant Courses

↪ Compilers – resulting Java compiler was most correct from class of 50 students

MISCELLANY

Interests

model checking, proof assistants, music, functional programming, compilers, robotics, electronics, math pedagogy