John Kelly

https://johnk.dev | johnharrykelly@gmail.com | linkedin.com/in/johnharrykelly | github.com/john-h-k

Education

Imperial College London London, England
Bachelor of Engineering in Electronic & Information Engineering (EECS) *Sep. 2023 - July 2026*

Experience

Quantitative Developer - Intern July. 2025 – Sep. 2025
Quadrature Capital *London, England*

- Incoming Summer 2025

Software Engineer - Intern July 2024 – Aug. 2024
Marshall Wace Asset Management *London, England*

- Worked in the data streaming team to develop an Elastic-based document search store system
- Utilised dense & sparse vector search algorithms to build fast search capabilities across different file formats, languages, and contexts
- Integrated search capabilities with internal tooling, ML models, and documentation systems

Software Engineer Sep. 2022 – Sep. 2023
Hero Health Software *Oxford, England*

- Led a multi-team project over 6 months, integrating a new clinical system into the application, spanning multiple languages, frameworks, and services, including Rust, GraphQL, ProtoBuf, and TypeScript
- Upgraded language & frameworks across 2 major versions, involving changes to over 500 files and 20 thousand lines of code
- Re-designed the CI/CD pipeline, cutting test times from 90 minutes to 15 minutes and reducing cost
- Optimised high-throughput data processing services, cutting median execution time from 2 hours to under 4 seconds

Software Engineer - Intern June 2021 – Aug. 2021
Hero Health Software *Oxford, England*

- Migrated a background-job service between frameworks
- Designed, implemented, and tested code & infrastructure for card-reader payment systems

Open Source Work
.NET JIT Compiler, Rust Compiler, ComputeSharp, & Others

- Contributed to compilers and low-latency open-source software in Rust, C++, & C#
- Small contributions to Roslyn (C# compiler), LLVM, Rust compiler, and Zig compiler (ongoing)

Achievements & Awards

ICHack 2025 Winner 2025
Helsing Challenge

Top 15 in Oxford University Computing Challenge 2019
Result out of over 10,000 international participants

First Place - OPTIC London Forecasting competition 2024

IBM Ponder Maths Puzzles 2023

Bebras Computational Thinking Challenge - Gold Award 2018

Microsoft Most Valuable Professional Award Nomination - Youngest Ever Nominee 2022

Perse Competitive Coding Cup - Distinction 2018

.NET Foundation Voting Member 2020

GCHQ CyberDiscovery Program 2019
Attended to the in-person CyberDiscovery Elite camp as part of the only team that successfully cracked the final challenge

Projects - *See 'Pinned' section of GitHub profile*

JCC | *C, Compilers, Optimisation* Nov. 2023 - Ongoing

- C11/17/23 compiler with zero 3rd party dependencies
- Fully bootstrapping (self compilation in under 1 second), and able to compile other large C projects such as SQLite
- Features optimisation layer with inlining, struct-promotion, dead code elimination, and strength reduction passes
- Pure C11 compliant code with a hand-written preprocessor, lexer, parser, and native x64 + ARM64 + RISC-V codegen backends
- Utilises SSA intermediate representation & linear-scan register allocation techniques for codegen

Rustf*ck (Brainf*ck interpreter + compiler) | *Rust, Compilers* June 2023

- High-performance Brainf*ck interpreter and JIT compiler
- Built parser & optimiser with 3-stage IR
- Developed an AArch64 Just-in-Time Compiler

MathSharp | *C#, x64, SIMD* Oct. 2019

- The fastest SIMD-focused linear algebra library for C# at time of release
- At time of release, offered 40-75% speed improvements over the .NET Core library & other alternatives
- Utilised x64 & AArch64 architecture extensions including FMA, SSE, AVX, and NEON
- Gained over 600 stars on Github & over 7,000 downloads

Voltium | *C++, C#, DirectX, Metal, 3D Graphics* Ongoing

- Lightweight, cross-platform render engine focusing on performance and usability
- Developed a system to allow remote rendering & debugging using a proprietary command buffer system
- Created a render graph & ECS framework to allow efficient scheduling & execution of rendering

Riscy | *Rust, RISC-V* Feb 2025

- Simple high-performance RISCV32 interpreter with 50-200x speedup over official RISC-V simulator
- Uses ELF section merging & instruction pre-caching to optimise hot-path memory accesses and decoding