Mikhail Razakov

SOFTWARE ENGINEER · COMPILERS, RUNTIME & PERFORMANCE

Moscow / remote · razakov.mikhail@outlook.com · Telegram · GitHub

PROFILE

I build compiler/runtime components in C#/.NET and C++/LLVM, from optimization-legality checks to generated-code validation. At ISP RAS I work on C# symbolic analysis in SharpChecker; at MCST I worked on LLVM/LICM. I keep correctness evidence and performance measurements separate for experimental optimizations.

TECHNICAL SKILLS

- C#/.NET, CIL/CLR, DynamicMethod, Reflection.Emit, Roslyn, BenchmarkDotNet; interpreter/CIL backends
- C++23/LLVM 22, IR/CFG/SSA, LICM, effect analysis, Rust/x86-64 (SysV ABI), differential testing, Git/Linux

EXPERIENCE

ISP RAS · SharpChecker Oct 2026 — present

Symbolic computation and structured C# code/declaration metadata for static analysis and AI-assisted defect recognition.

MCST · compiler engineering internship Jul — Aug 2026

C++23/LLVM 22: experimental LICM with conservative legality checks; Python differential checks of before/after behavior.

Commercial development · C#/.NET & adjacent technologies approx. 1 year total

Application and backend work with C#/.NET, Unity, ASP.NET Core, SQL/PostgreSQL, file I/O and servers. Duration is candidate-reported.

PROJECTS

UniversalToolchain · compiler/runtime infrastructure

Module contracts and dependencies are checked in LanguagePlan before execution. Separate interpreter/CIL backends allow result comparisons for the same language, with dedicated parity scenarios.

LLVM Global IV / LICM · optimization legality

Interprocedural effect analysis is separated from PromotionLegality and the IR transformation. Volatile/atomic accesses, unknown calls and unsupported CFG conservatively reject optimization; negative regression cases cover unsafe paths.

x86-64 codegen · Rust

SSA/CFG → liveness → register allocation → x86-64 pipeline. An independent verifier checks register assignments and spills; native execution is compared against a reference IR interpreter.

Quark2 · generated-code invocation measurements

DynamicMethod/Reflection.Emit and RuntimeMethodHandle; BenchmarkDotNet compares reflection Invoke, delegate and pointer calls on generated methods.

EDUCATION AND RECOGNITION

HSE University — Software Engineering.

MEPhI Junior 2026, 1st-degree diploma; LangDev 2026 speaker.