How to reduce the size of Stylus WASM binaries

ALPHA RELEASE, PUBLIC PREVIEW DOCS Stylus is currently tagged as analpha release. The code has not been audited, and shouldnot be used in production scenarios . This documentation is currently inpublic preview .

To provide feedback, click theRequest an update button at the top of this documentjoin the Arbitrum Discord , or reach out to our team directly by completingthis form . To be deployed onchain, the size of youruncompressed WebAssembly (WASM) file must not exceed 128Kb, while thecompressed binary must not exceed 24KB. Stylus conforms with the same contract size limit as the EVM to remain fully interoperable with all smart contracts on Arbitrum chains.

TheStylus CLI automatically compresses your WASM programs, but there are additional steps that you can take to further reduce the size of your binaries. Your options fall into two categories: Rust compiler flags, and third-party optimization tools.

Rust compiler flags

The Rust compiler supports various config options for shrinking binary sizes.

Cargo.toml

[profile.release] codegen-units = 1 # prefer efficiency to compile time panic = "abort" # use simple panics opt-level = "z" # optimize for size ("s" may also work) strip = true # remove debug info lto = true # link time optimization debug = false # no debug data rpath = false # no run-time search path debug-assertions = false # prune debug assertions incremental = false # no incremental builds

.cargo/config.toml

[build] target = "wasm32-unknown-unknown"

[target.wasm32-unknown-unknown] rustflags = ["-C", "link-arg=-zstack-size=8192", # shrink the heap]

nightly flags

Additional unstable nightly flags may help too.

cargo +nightly build -Z build-std

std,panic_abort -Z build-std-features

panic_immediate_abort The above configures the standard library to not include panic strings, which are useless on chain but may take up considerable space.

cargo stylus deploy --nightly

Third-party optimization tooling

Additional WASM-specific tooling exists to shrink binaries. Due to being 3rd party, users should use these at their own risk.

wasm-opt

wasm-opt applies techniques to further reduce binary size, usually netting around 10%.

Frequently asked questions

Will future releases of Stylus introduce additional optimizations?

Yes! We're actively working on improving WASM sizes generated by Rust code with the Stylus SDK.

Why don't I have to worry about this type of optimization when I usecargo without using Stylus?

On modern platforms, tools likecargo don’t have to worry about the size of the binaries they produce. This is because there’s many orders of magnitude more storage available than even the largest of binaries, and for most applications it’s media like images and videos that constitutes the majority of the footprint.

Resource constraints when building on blockchains are extremely strict. Hence, while not the default option, tooling often provides mechanisms for reducing binary bloat, such as the options outlined in this document. Edit this page Last updatedonMar 7, 2024 Previous How to add a new programming language to StylusNext Public preview: What to expect