

# Basic instructions

To compile release version of the smart contract you can run:

`cargo build --target wasm32-unknown-unknown --release` info The above build command is setting a target flag to create a WebAssembly.wasm file. Notice that your project directory now has a few additional items:

. |—— Cargo.lock ← created during build to lock dependencies |—— Cargo.toml |—— src | |—— lib.rs |——  
target ← created during build, holds the compiled wasm

## Build and Flags

We recommend you to optimize your build artifact with the use of the next flags in your Cargo.toml file. If you are performing a multi-contract build, you should include these settings in the Cargo.toml that is at the root of your project.

`[profile.release] codegen-units = 1`

## Tell rustc to optimize for small code size.

`opt-level = "z" lto = true debug = false panic = "abort"`

## Opt into extra safety checks on arithmetic operations

<https://stackoverflow.com/a/64136471/249801>

`overflow-checks = true` The above command is essentially setting special flags and optimizing the resulting.wasm file. At the end of the day, this allows you to customize the `cargo build --release` command.

## Custom Flags

If you wish to add custom flags to your build, you can perform this by adding build flags to your `ProjectFolder/.cargo/config.toml` as illustrated in this example.

`[target.wasm32-unknown-unknown] rustflags = ["-C", "link-arg=-s"]` A full set of build options can be accessed at <https://doc.rust-lang.org/cargo/reference/config.html>.

You can find an example [here](#). [Edit this page](#) Last updated on Aug 24, 2022 by Damián Parrino Was this page helpful? Yes No

[Previous Deploying Contracts](#) [Next Rapid Prototyping](#)