# rustfmt Rust code formatter

## CONCEPTS

cargo fmt reads the edition from Cargo.toml; bare rustfmt assumes 2015 unless told otherwise.

**Source files** *.rs
↓ parse (edition)
**rustfmt** rustfmt.toml + style_edition
↓ emit
**Result** files, stdout or diff

edition controls parsing; style_edition controls the formatting rules. Set both in rustfmt.toml.

Unstable options need nightly plus unstable_features = true; stable rustfmt ignores them with a warning.

## INSTALL AND RUN

**rustup component add rustfmt**
Install on the stable toolchain

**rustup component add rustfmt --toolchain nightly**
Install on nightly

**cargo fmt**
Format all targets of the crate

**cargo fmt --all**
Format every workspace member

**cargo fmt -p** *<crate>*
Format one package

**cargo fmt --** *<rustfmt-args>*
Pass flags through to rustfmt

**cargo +nightly fmt**
Run the nightly formatter

**rustfmt** *<file.rs>*
Format a file and its out-of-line modules

**rustfmt** *<a.rs> <b.rs>*
Format several files in place

**echo 'fn main(){}' | rustfmt**
Format stdin to stdout

**rustfmt --version**
Show the rustfmt version

**rustfmt --help=config**
List config options and defaults

## CHECKING AND OUTPUT

**cargo fmt --check**
Exit 1 and print a diff if unformatted

**cargo fmt --all -- --check**
Check the whole workspace in CI

**rustfmt --check** *<file.rs>*
Check one file without writing

**rustfmt --emit stdout** *<file.rs>*
Print the result instead of writing

**rustfmt -l** *<file.rs>*
List files that would change

Exit codes: --check returns 1 on needed changes; other modes return 1 only on errors (parse failure).

## CONFIGURATION

**rustfmt.toml / .rustfmt.toml**
Config file in the project or any parent dir

**rustfmt --print-config default rustfmt.toml**
Write all defaults to a file

**rustfmt --print-config current** *<file.rs>*
Show the effective config for a file

**rustfmt --print-config minimal rustfmt.toml**
Write only non-default values

**rustfmt --config-path** *<dir|file>* *<file.rs>*
Use a specific config

**rustfmt --config max_width=80** *<file.rs>*
Override an option from the CLI

**cargo fmt -- --config max_width=80,tab_spaces=2**
Override options via cargo

**edition = "2021"**
Parse with this language edition

**style_edition = "2024"**
Pick the style guide edition

**rustfmt --edition 2021** *<file.rs>*
Set the edition on the command line

## CONFIGURATION (cont.)

**rustfmt --style-edition 2024** *<file.rs>*
Set the style edition on the CLI

Set edition and style_edition in rustfmt.toml so cargo fmt and plain rustfmt agree.

## COMMON OPTIONS (STABLE)

**max_width = 100**
Maximum line width

**tab_spaces = 4**
Spaces per indent level

**hard_tabs = false**
Indent with tabs instead of spaces

**use_small_heuristics = "Default"**
Scale small-width limits: Off, Max, Default

## NIGHTLY-ONLY OPTIONS

**unstable_features = true**
Opt in to unstable options

**imports_granularity = "Crate"**
Merge imports: Preserve, Crate, Module, Item, One

**group_imports = "StdExternalCrate"**
Group imports: Preserve, StdExternalCrate, One

**wrap_comments = true**
Re-wrap long comments

**format_code_in_doc_comments**
Format Rust code blocks in docs

**ignore = ["*<path>*"]**
Skip files/dirs (gitignore patterns)

Only some options are stable; the rest print a warning and are skipped on a stable toolchain.

## SKIPPING AND ATTRIBUTES

**#[rustfmt::skip]**
Do not format the next item or statement