TypeScript compiler (tsc) Typed JS compiler

tsconfig.json options + files → tsc → **Type check** errors, exit code 1 → emit → **JavaScript** outDir

CONCEPTS

tsc finds tsconfig.json by walking up from the current directory. Naming files on the command line makes it refuse to run unless you add --ignoreConfig.

Type errors do not stop emit unless noEmitOnError is set. A non-zero exit code only tells you there were errors.

These entries describe TypeScript 7 (native compiler). 7 removed baseUrl, target es5 and moduleResolution node10, and made strict the default. On 5.x, check with tsc --version.

PROJECT SETUP

`tsc --init`
Create a tsconfig.json

`tsc`
Compile using the nearest tsconfig.json

`tsc -p <path>`
Compile a config file or directory

PROJECT SETUP (cont.)

`tsc --noEmit`
Type-check without writing files

`tsc --watch`
Recompile when files change

`tsc --ignoreConfig <file>`
Compile given files, ignore tsconfig

`tsc --version`
Show the compiler version

`tsc --help --all`
List every compiler option

TSCONFIG KEYS

`"extends": "<file>"`
Inherit another config

`"include": ["src"]`
Files to compile

`"exclude": ["dist"]`
Files to skip

`"files": ["a.ts"]`
Explicit file list

`"references": [{"path": "<dir>"}]`
Link another project

`"compilerOptions": {}`
Where all options below go

LANGUAGE & MODULES

`--target esnext`
Syntax level of emitted JS

`--module nodenext`
Module format for Node projects

`--module preserve`
Keep imports as written (bundlers)

`--moduleResolution bundler`
Resolution for bundler setups

`--lib <list>`
Built-in type libraries to include

`--types <list>`
Which @types packages to load

`--jsx react-jsx`
How JSX is compiled

`--resolveJsonModule`
Allow importing .json files

`--allowJs`
Include .js files in the project

`--checkJs`
Report errors in .js files

`"paths": {"@/*": ["./src/*"]}`
Alias import prefixes

TYPE CHECKING

`--strict`
Enable all strict checks

`--noUncheckedIndexedAccess`
Indexed access may be undefined

`--noImplicitReturns`
Every code path must return

`--noUnusedLocals`
Error on unused variables

`--noUnusedParameters`
Error on unused parameters

`--skipLibCheck`
Skip checking .d.ts files

EMIT

`--outDir <dir>`
Write output files here

`--rootDir <dir>`
Root of the input tree

`--declaration`
Emit .d.ts files

`--emitDeclarationOnly`
Emit only .d.ts files

`--sourceMap`
Emit .js.map files

`--noEmitOnError`
Emit nothing if there are errors

EMIT (cont.)

`--verbatimModuleSyntax`
Keep imports/exports as written

`--isolatedModules`
Require per-file safe code

PROJECTS & BUILDS

`tsc --build`
Build projects whose inputs changed

`tsc -b --watch`
Rebuild referenced projects on change

`tsc -b --clean`
Delete build outputs

`--incremental`
Cache state in .tsbuildinfo

DEBUGGING

`tsc --showConfig`
Print the final merged config

`tsc --listFiles`
List every file in the program

`tsc --listFilesOnly`
List files and stop

`tsc --explainFiles`
Why each file was included

`tsc --traceResolution`
Log module lookup steps