Rollup JS module bundler

CONCEPTS

> Rollup bundles ES modules; resolving npm packages or CommonJS needs plugins (node-resolve, commonjs).

Entry input
↓ resolve + load
Module graph plugins transform
↓ tree-shake
Chunks only used code
↓ render format
Output file or dir

> Tree-shaking works on static import/export; mark pure calls with /*#__PURE__*/ and set sideEffects in package.json.

BUILD

`rollup -c`
Build using rollup.config.(mjs/cjs/js)

`rollup -c <file>`
Build using a named config

`rollup -i <entry> -o <file>`
Bundle entry into one file

`rollup -i <entry> -d <dir>`
Write chunks to a directory

`rollup -i <entry> -o <file> -f <fmt>`
Pick output format

`rollup -c -w`
Watch and rebuild

`rollup -c --failAfterWarnings`
Fail the build on warnings

`rollup -v`
Show version

OUTPUT OPTIONS

`-f iife -n <Name>`
Browser global with name

`-f umd -n <Name>`
Universal module, needs a name

OUTPUT OPTIONS (cont.)

`-g <id>:<Global>`
Map external to global (iife/umd)

`-m`
Write sourcemap file

`-m inline`
Embed the sourcemap in the bundle

`--exports <mode>`
Set export mode

`--entryFileNames <pattern>`
Name entry chunks

DEPENDENCIES

`-e <ids>`
Keep modules out of the bundle

`--inlineDynamicImports`
Fold dynamic imports into one file

`--preserveModules`
One output file per module

PLUGINS AND CONFIG

`-p <plugin>`
Use a plugin by name

`--environment <KEY[:val]>`
Set process.env values for the config

`--configPlugin <plugin>`
Transpile the config file (TypeScript)

TREE-SHAKING

`--no-treeshake`
Turn tree-shaking off

`--treeshake.manualPureFunctions <names>`
Declare functions as pure

`--no-treeshake.moduleSideEffects`
Assume imports have no side effects

WATCH MODE

`-w`
Watch files, rebuild on change