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.

CLI flags override the same options in the config file.

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 <entry> -o <file>`
Same, entry as plain argument

`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

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

OUTPUT OPTIONS (cont.)

`-m`
Write sourcemap file

`-m inline`
Embed the sourcemap in the bundle

`--exports <mode>`
Set export mode

`--entryFileNames <pattern>`
Name entry chunks

`--chunkFileNames <pattern>`
Name shared chunks

`--banner <text>`
Text at the top of the bundle

DEPENDENCIES

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

`--inlineDynamicImports`
Fold dynamic imports into one file

`--preserveModules`
One output file per module

`--preserveModulesRoot <path>`
Strip this prefix from module paths

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)

`--bundleConfigAsCjs`
Allow __dirname and require in config

`--config<Name>`
Custom flag read by a config function

TREE-SHAKING

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

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

TREE-SHAKING (cont.)

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

WATCH MODE

`-w`
Watch files, rebuild on change

`--watch.exclude <files>`
Do not watch these files

`--no-watch.clearScreen`
Keep the terminal on rebuild

LOGGING AND DEBUGGING

`--logLevel <level>`
Choose which logs show

`--filterLogs <filter>`
Show only matching logs

`--perf`
Print timing per phase

RECIPES

Library with ES and CJS builds

```
# rollup.config.mjs
import { defineConfig } from
'rollup';
export default defineConfig({
  input: 'src/index.js',
  external: ['react'],
  output: [
    { file: 'dist/index.mjs',
format: 'es', sourcemap: true },
    { file: 'dist/index.cjs',
format: 'cjs', sourcemap: true }
  ]
});
```

Browser bundle with resolve and commonjs

```
npm i -D rollup @rollup/plugin-
node-resolve @rollup/plugin-
commonjs
# in the config: plugins:
[resolve(), commonjs()]
rollup -c --environment
BUILD:production
```

TypeScript config file

```
npm i -D @rollup/plugin-
typescript
rollup -c rollup.config.ts --
configPlugin typescript
```

Quick IIFE for a script tag

```
rollup -i src/main.js -o dist/
app.js -f iife -n App \
  -g jquery:jQuery -e jquery -m
```