esbuild Fast JS bundler

CONCEPTS

Without --bundle esbuild only transforms each file (TS/JSX to JS) and does not follow imports. With no --outfile it prints to stdout.

Entry points app.ts, ...
↓ resolve
Module graph imports, packages
↓ loader + target
Transformed TS/JSX to JS, minify
↓ format
Output --outfile / --outdir

Flag forms: --flag, --flag=value for single values, --flag:key=value for repeatable ones (--loader:.png=file, --external:react).

BUILD

esbuild *<entry>* **--bundle --outfile=***<file>*
Bundle one entry into a file

esbuild *<a>* *<b>* **--bundle --outdir=***<dir>*
Bundle several entries into a directory

--minify
Minify whitespace, syntax and names

--sourcemap
Write a linked .map file

--sourcemap=inline
Embed the source map as a data URL

--tsconfig=*<file>*
Use this tsconfig instead of discovery

PLATFORM AND FORMAT

--platform=browser|node|neutral
Pick target environment (default browser)

--format=iife|cjs|esm
Output module format

PLATFORM AND FORMAT (cont.)

--target=es2020,chrome100
Lower syntax for these engines

--external:*<pkg>*
Leave a module out of the bundle

--packages=external
Keep every package import external

--global-name=*<name>*
Global variable name for iife output

LOADERS AND ASSETS

--loader:.*<ext>***=***<loader>*
Choose how a file type is read

--loader:.png=file
Copy asset and import its path

--jsx=automatic
Use the React 17+ JSX runtime

OPTIMIZE

--define:*<name>***=***<value>*
Replace an identifier with a constant

--define:process.env.NODE_ENV=\"production\"
Set NODE_ENV for dead-code removal

WATCH AND SERVE

--watch
Rebuild when inputs change

--servedir=*<dir>*
Serve a directory plus build output

ANALYZE AND LOGS

--metafile=*<file>*
Write build metadata JSON

--analyze
Print a size report of the bundle

TRANSFORM STDIN

esbuild --loader=ts < *<in.ts>*
Transform TypeScript from stdin