@babel/cli

EDIT

Babel comes with a built-in CLI which can be used to compile files from the command line.

In addition, various entry point scripts live in the top-level package at `@babel/cli/bin` . There is a shell-executable utility script, `babel-external-helpers.js` , and the main Babel cli script, `babel.js` .

Install

While you *can* install Babel CLI globally on your machine, it's much better to install it **locally** project by project.

There are two primary reasons for this.

1. Different projects on the same machine can depend on different versions of Babel allowing you to update them individually.
2. Not having an implicit dependency on the environment you are working in makes your project far more portable and easier to setup.

We can install Babel CLI locally by running:

Shell Copy

```
npm install --save-dev @babel/core @babel/cli
```

> **Note:** If you do not have a `package.json` , create one before installing. This will ensure proper interaction with the `npx` command.

After that finishes installing, your `package.json` file should include:

Diff Copy

```
{
  "devDependencies": {
```

Usage

> **Note:** Please install `@babel/cli` and `@babel/core` first before `npx babel`, otherwise `npx` will install out-of-dated `babel` 6.x. Other than npx, you can also drop it inside of an npm run script or you may instead execute with the relative path instead. `./node_modules/.bin/babel`

Shell Copy

```
npx babel script.js
```

Compile Files

Compile the file `script.js` and **output to stdout**.

Shell Copy

```
npx babel script.js
# output...
```

If you would like to **output to a file** you may use `--out-file` or `-o`.

Shell Copy

```
npx babel script.js --out-file script-compiled.js
```

To compile a file **every time that you change it**, use the `--watch` or `-w` option:

Shell Copy

```
npx babel script.js --watch --out-file script-compiled.js
```

Compile with Source Maps

If you would then like to add a **source map file** you can use `--source-maps` or `-s`. Le

Shell Copy

Shell Copy

```
npx babel script.js --out-file script-compiled.js --source-maps inline
```

Compile Directories

Compile the entire `src` directory and output it to the `lib` directory by using either `--out-dir` or `-d`. This doesn't overwrite any other files or directories in `lib`.

Shell Copy

```
npx babel src --out-dir lib
```

Compile the entire `src` directory and output it as a single concatenated file.

Shell Copy

```
npx babel src --out-file script-compiled.js
```

Ignore files

Ignore spec and test files

Shell Copy

```
npx babel src --out-dir lib --ignore "src/**/*.spec.js","src/**/*.test.js"
```

Copy files

Copy files that will not be compiled

Shell Copy

```
npx babel src --out-dir lib --copy-files
```

If you don't want to copy ignored JavaScript files:

Shell Copy

Pipe a file in via stdin and output it to `script-compiled.js`

Shell Copy

```
npx babel --out-file script-compiled.js < script.js
```

Using Plugins

Use the `--plugins` option to specify plugins to use in compilation

Shell Copy

```
npx babel script.js --out-file script-compiled.js --plugins=@babel/proposal-
class-properties,@babel/transform-modules-amd
```

Using Presets

Use the `--presets` option to specify presets to use in compilation

Shell Copy

```
npx babel script.js --out-file script-compiled.js --presets=@babel/preset-
env,@babel/flow
```

Ignoring .babelrc.json or .babelrc

Ignore the configuration from the project's `.babelrc` or `.babelrc.json` file and use the cli options e.g. for a custom build

Shell Copy

```
npx babel --no-babelrc script.js --out-file script-compiled.js --
presets=@babel/preset-env,@babel/preset-react
```

Custom config path

Shell Copy

Added in: `v7.8.0`

By default, Babel will override the extension of the transpiled file and use `.js` instead.

To preserve the original file extension you can pass the `--keep-file-extension`.

You can also control what file extension is used with `--out-file-extension .example-extension` e.g. `babel src/ lib/ --out-file-extension .mjs`.

Note that `--keep-file-extension` and `--out-file-extension` cannot be used together.

Advanced Usage

There are many more options available, see options, `babel --help` and other sections for more information.

← EDITORS

@BABEL/POLYFILL →