babel-compiler

Source code of released version | Source code of development version

---

Babel is a parser and transpiler for ECMAScript 2015 syntax and beyond, which enables some upcoming JavaScript syntax features to be used in today's browsers and runtimes.

Meteor's Babel support consists of the following core packages:

- `babel-compiler` - Exposes the Babel API on the symbol `Babel`. For example, `Babel.compile(source, options)`.
- `babel-runtime` - Meteor versions of the external helpers used by Babel-generated code. Meteor's core packages must run on IE 8 without polyfills, so these helpers cannot assume the existence of `Object.defineProperty`, `Object.freeze`, and so on.

Babel API

The `babel-compiler` package exports the `Babel` symbol, which exposes functionality provided by the `@meteorjs/babel` NPM package, which is in turn implmented using the `babel-core` NPM package. Note that you can only use the `babel-compiler` package on the server.

Example:

```
var babelOptions = Babel.getDefaultOptions();

// Modify the default options, if necessary:
babelOptions.whitelist = [
  "es6.blockScoping", // For `let`
  "es6.arrowFunctions" // For `=>`
];

var result = Babel.compile(
  "let square = (x) => x*x;",
  babelOptions
);

// result.code will be something like
// "var square = function (x) {\n  return x * x;\n};"
```

Use `Babel.compile(source)` to transpile code using a set of default options that work well for Meteor code.

`.babelrc` configuration files

Like other Babel-compiled projects, a Meteor project that uses the `ecmascript` package can specify custom Babel plugins and presets (which are just groups of plugins) in JSON files named `.babelrc`.

For example, to enable the Babel transform that supports class properties, you should

1. run `meteor npm install --save-dev babel-plugin-transform-class-properties`
2. put the following in a `.babelrc` file in the root of your project:

```
{
  "plugins": ["transform-class-properties"]
}
```

If you want to include all Stage 1 transforms (including the class properties plugin), you could use a preset instead:

```
meteor npm install --save-dev babel-preset-stage-1
```

and then (in your `.babelrc` file):

```
{
  "presets": ["stage-1"]
}
```

Note that you should never need to include the `es2015` or `react` transforms, as that functionality is already provided by the default `babel-preset-meteor` preset.

Any plugins and transforms that you list in your `.babelrc` file will be included after `babel-preset-meteor`.

To be considered by the `babel-compiler` package, `.babelrc` files must be contained within your root application directory.

Resources:

- API docs
- List of transformers