Using ES modules in Node

Node has had experimental support for EcmaScript modules for a while.

Again, this is one thing I've wanted to explore for a while but have never felt the need to dig deeply. After all, it is experimental and it hasn't been approved for production use.

But wit the release of Node 14 we're coming to the point when module support moved to stable.

So let's explore what it takes to run ES Modules in Node on their own and together with current Node modules.

What will Node consider an ES Module?

There are certain file extensions and conditions that will cause Node to treat files as ES Modules. These conditions/extensions include:

- Files ending in `.mjs`
- Files ending in `.js` when the nearest parent `package.json` file contains a top-level field `"type": "module"`
- Files ending in `.js` when Node runs with the `--experimental-modules` flag (versions of Node before 14.x)
- Strings passed in as an argument to `--eval` or `--print`, or piped to node via `STDIN`, with the flag `--input-type=module`

So, in theory, we could use `.mjs` for all our ES Modules files but we need to be careful as your server needs to be confiured to serve `.mjs` files as Javascript and I'm not certain all servers are confiured to do this out of the box.

File extensions

Two extensions have special meaning. As we discussed earlier in this post, `.mjs` will always be treated as an ES Module file

```
import 'commonjs-package/src/index.mjs';
// Loaded as ES module since .mjs is always
// loaded as ES module.
```

The `.cjs` extension is the opposite. This file will always load as a Common JS module.

```
import './legacy-file.cjs';
// Loaded as CommonJS since .cjs is always
// loaded as CommonJS.
```

Running Node with module support

As of Node 12.16.3, the latest LTS Node version as of this writing, the command to run Node with experimental Node Module support is:

```
node --experimental-modules colors.js
```

Using this flag we'll be able to run the same code both on Node and on the browser.

If you're interested, keep an eye for anouncements about modules in Node. This may change in unpexpected ways so check the documentation and be ready.

Links and Resources

- How to use ECMAScript modules with Node.js (LogRocket)
- JavaScript modules (V8)
- The new ECMAScript module support in Node.js 12
- Node.js v12.16.3 Documentation for ES Modules