JavaScript Module Systems

some demo stuff:
github.com/jmsv/js-module-systems

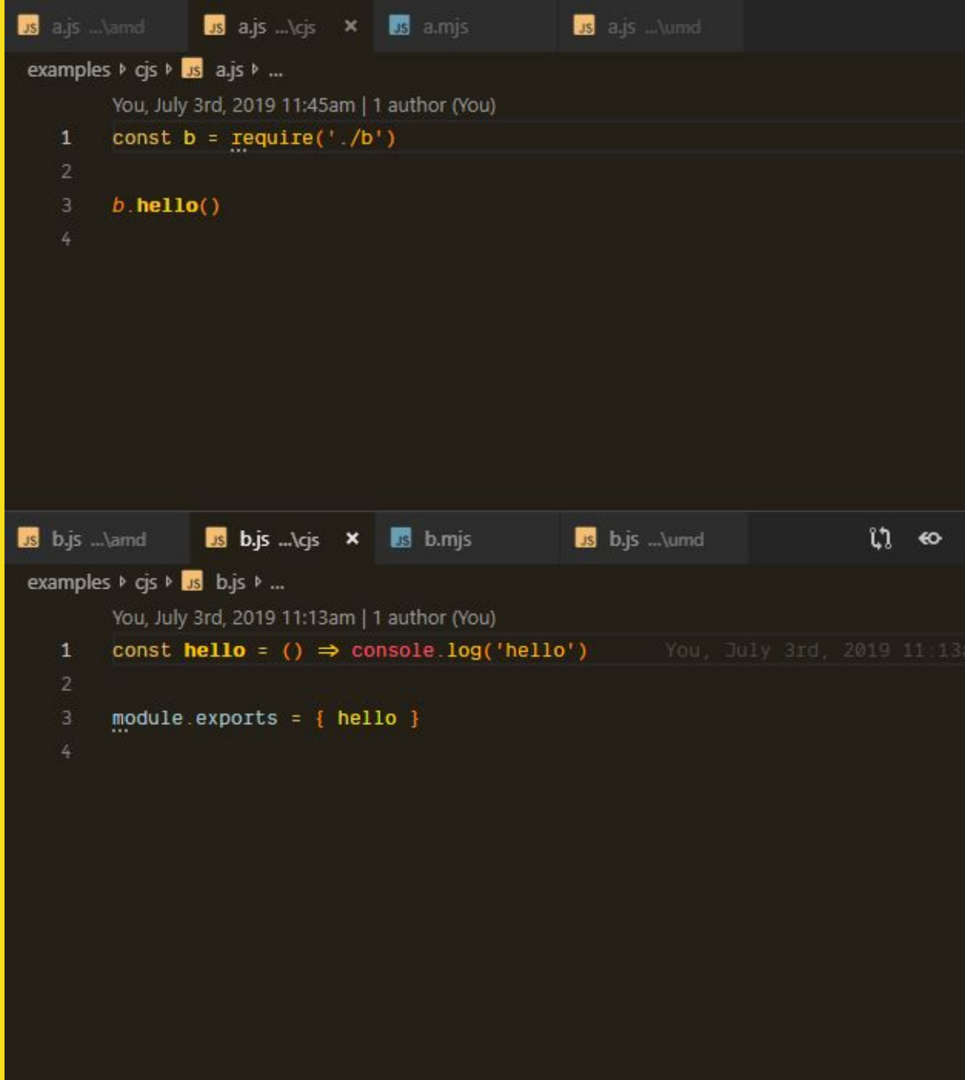
module types that imma cover

- CommonJS
- AMD
- UMD
- ES6 Modules

server-side

CommonJS

- Used by Node
- `require` imports an object


a.js ...\amd
a.js ...\cjs
a.mjs
a.js ...\umd
examples › cjs › a.js › ...
You, July 3rd, 2019 11:45am | 1 author (You)

```
const b = require('./b')

b.hello()

```

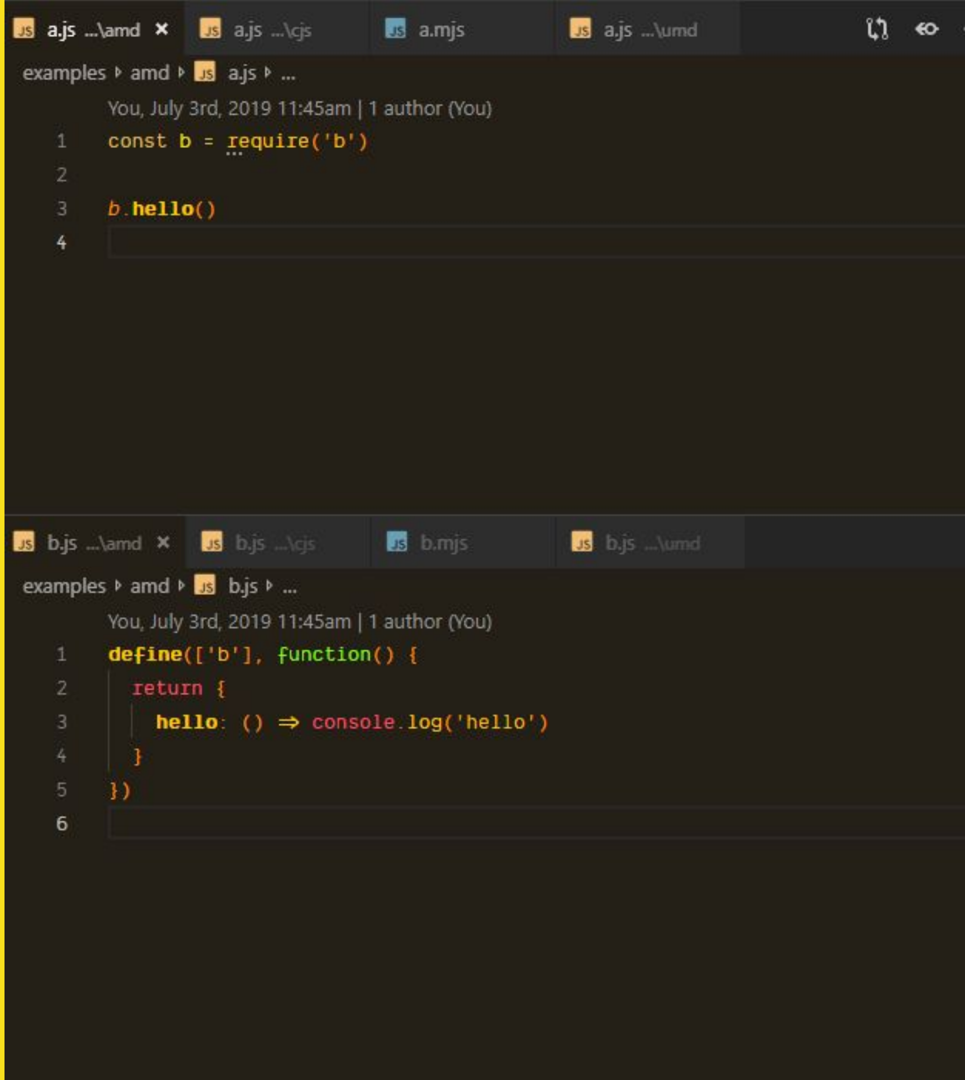
b.js ...\amd
b.js ...\cjs
b.mjs
b.js ...\umd
examples › cjs › b.js › ...
You, July 3rd, 2019 11:13am | 1 author (You)
You, July 3rd, 2019 11:13

```
const hello = () => console.log('hello')

module.exports = { hello }

```

client-side

AMD

- Used by RequireJS
- Implements `require` client-side



You, July 3rd, 2019 11:45am | 1 author (You)

```js
const b = require('b')

b.hello()

```

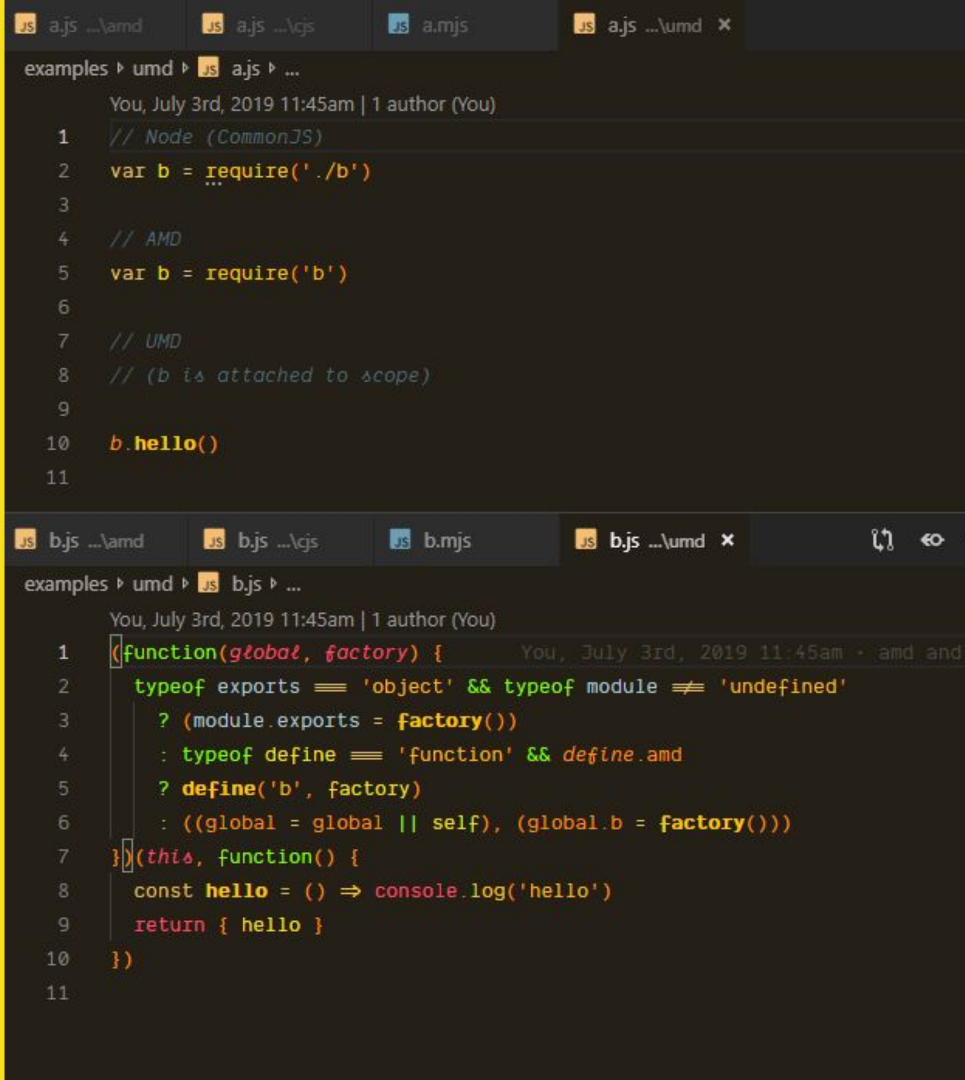
You, July 3rd, 2019 11:45am | 1 author (You)

```js
define(['b'], function() {
  return {
    hello: () => console.log('hello')
  }
})

```

server-side & client-side

UMD

- Combination of CommonJS and AMD
- Also attaches to global object



You, July 3rd, 2019 11:45am | 1 author (You)

```js
// Node (CommonJS)
var b = require('./b')

// AMD
var b = require('b')

// UMD
// (b is attached to scope)

b.hello()

```

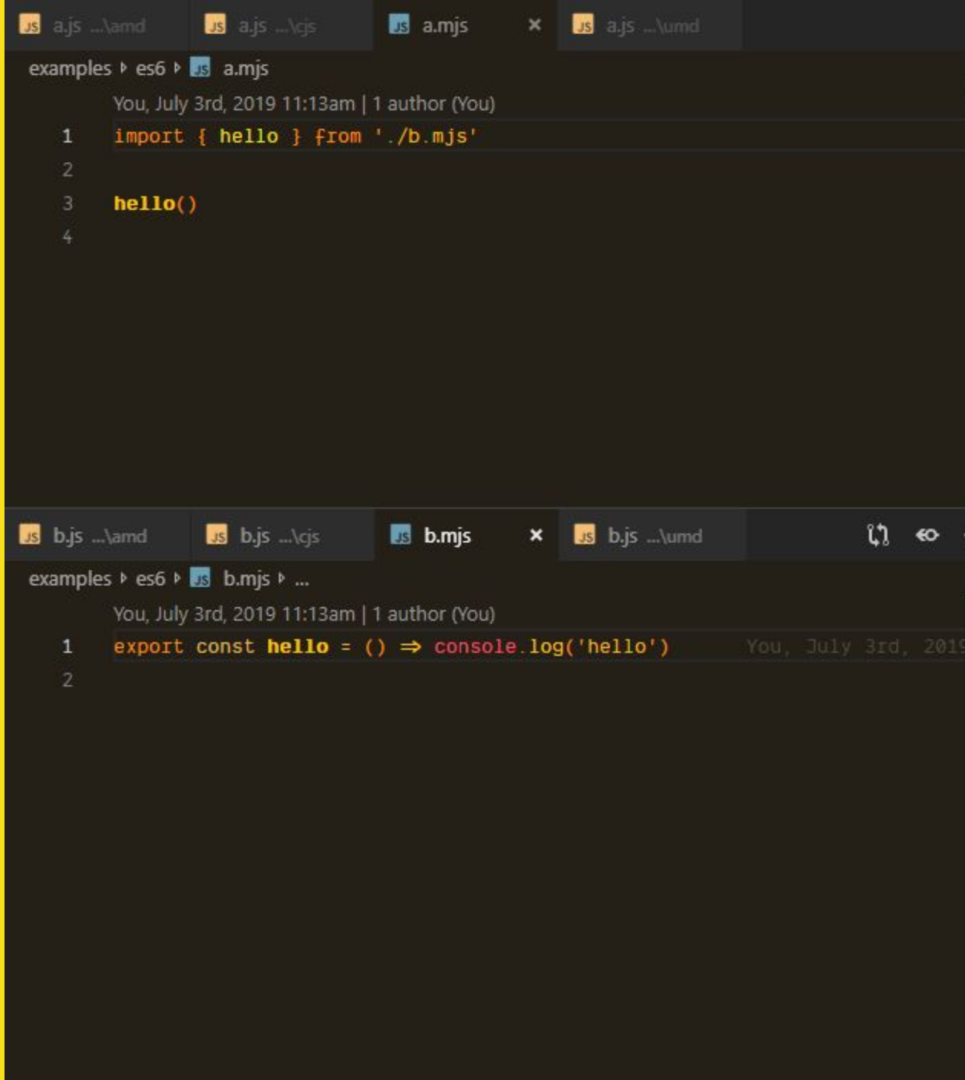
You, July 3rd, 2019 11:45am | 1 author (You)

```js
(function(global, factory) {          You, July 3rd, 2019 11:45am • amd and
  typeof exports === 'object' && typeof module !== 'undefined'
    ? (module.exports = factory())
    : typeof define === 'function' && define.amd
    ? define('b', factory)
    : ((global = global || self), (global.b = factory()))
})(this, function() {
  const hello = () => console.log('hello')
  return { hello }
})

```

server-side & client-side

ES6 Modules

- `import` gets binding values, not an object
- Static analysis enables tree shaking



a.js ...\amd
a.js ...\cjs
a.mjs
a.js ...\umd
examples › es6 › a.mjs
You, July 3rd, 2019 11:13am | 1 author (You)

```
import { hello } from './b.mjs'

hello()

```

b.js ...\amd
b.js ...\cjs
b.mjs
b.js ...\umd
examples › es6 › b.mjs › ...
You, July 3rd, 2019 11:13am | 1 author (You)

```
export const hello = () => console.log('hello')        You, July 3rd, 2019

```

bindings vs values

- CommonJS `require()` loads copies of values from the module being required

- ES6 import defines bindings to the actual values in the module definition

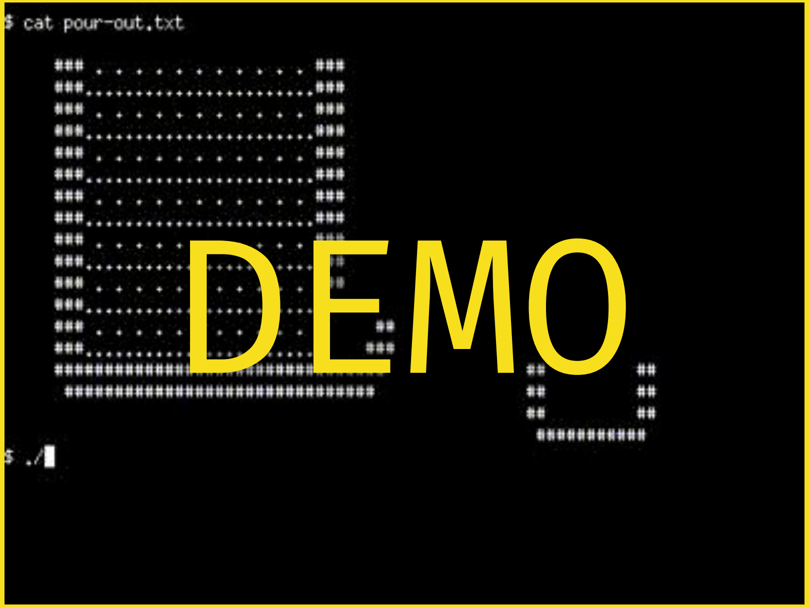
```
export var hi = 'hello'
setTimeout(() => hi = 'bye', 1000)
```

^ pls don’t do this tho lol

tree shaking

- Made possible by ES6 supporting static analysis
- Objects that aren’t necessary are omitted from the bundle

reference: bitsofco.de/what-is-tree-shaking



```
$ cat pour-out.txt

     ### . . . . . . . . . . . ###
     ###.......................###
     ### . . . . . . . . . . . ###
     ###.......................###
     ### . . . . . . . . . . . ###
     ###.......................###
     ### . . . . . . . . . . . ###
     ###.......................###
     ### . . . . . . . . . . . ###
     ###.......................###
     ### . . . . . . . . . . . ###
     ###.......................###
     ### . . . . . . . . . . .       ##
     ###.......................     ###
     #################################               ##         ##
      ###############################                ##         ##
                                                     ##         ##
                                                      ###########

$ ./
```

references

- https://www.freecodecamp.org/news/anatomy-of-js-module-systems-and-building-libraries-fadcd8dbd0e
- https://ponyfoo.com/articles/es6-modules-in-depth
- https://bitsofco.de/what-is-tree-shaking
- https://webpack.js.org/guides/tree-shaking