18. how Require() modules work

Monday, June 12, 2023 4:19 PM

-Each JS file is treated as a separate module;

-Node.js uses **CommonJS module system**:
require()
exports
module.exports

-**ES module system** used in browsers: import/export;

-Native ES modules (.mjs):
Not popular using .mjs

require('module');
Where does it come from & How does it work behind the scenes?

1. Resolving & Loading:
3 types of modules (as mentioned)
i. Core modules:
require('http');

ii. Developer mdules:
require('./lib/controller');

iii. 3rd-party modules (from NPM):
require('express');

Mechanism:
Path Resolving: **How Node decides which module to load**

1. Start with **core modules**
2. If begins with '**./**' or '**../**' => Try to **load developer modules**;
3. If no file found => Try to **find folder** with index.js in it;
4. Else => Go to **node_modules/** & try to find module there

2. Wrapping:
require();
// Like global variables injected

// Node puts require() into Immediately Invoked Functional Expression (IIFE) that returns objects

```
(function(exports, require, module, __filename, __dirname) {
	// Module code lives here...
});
```

// This keeps 'top-level' variables we defined in our modules private
// Scoped only to current module
// will NOT mess up with npm modules

require: function to require modules;
module: reference to the current module;
exports: a reference to module.exports, used to export obj from a module;
__filename: absolute path of the current module's file;
__dirname: directory name of the current module (current dir)

3. Execution:
Where the modules get executed by Node.js runtime

4. Returning exports:
-require function in our module => required module **exports**
-module.exports is the returned object (in required modules)

-module.exports is the returned object (in required modules)
-Use module.exports to export 1 single variable
i.e.
(-module.exports = Calculator);

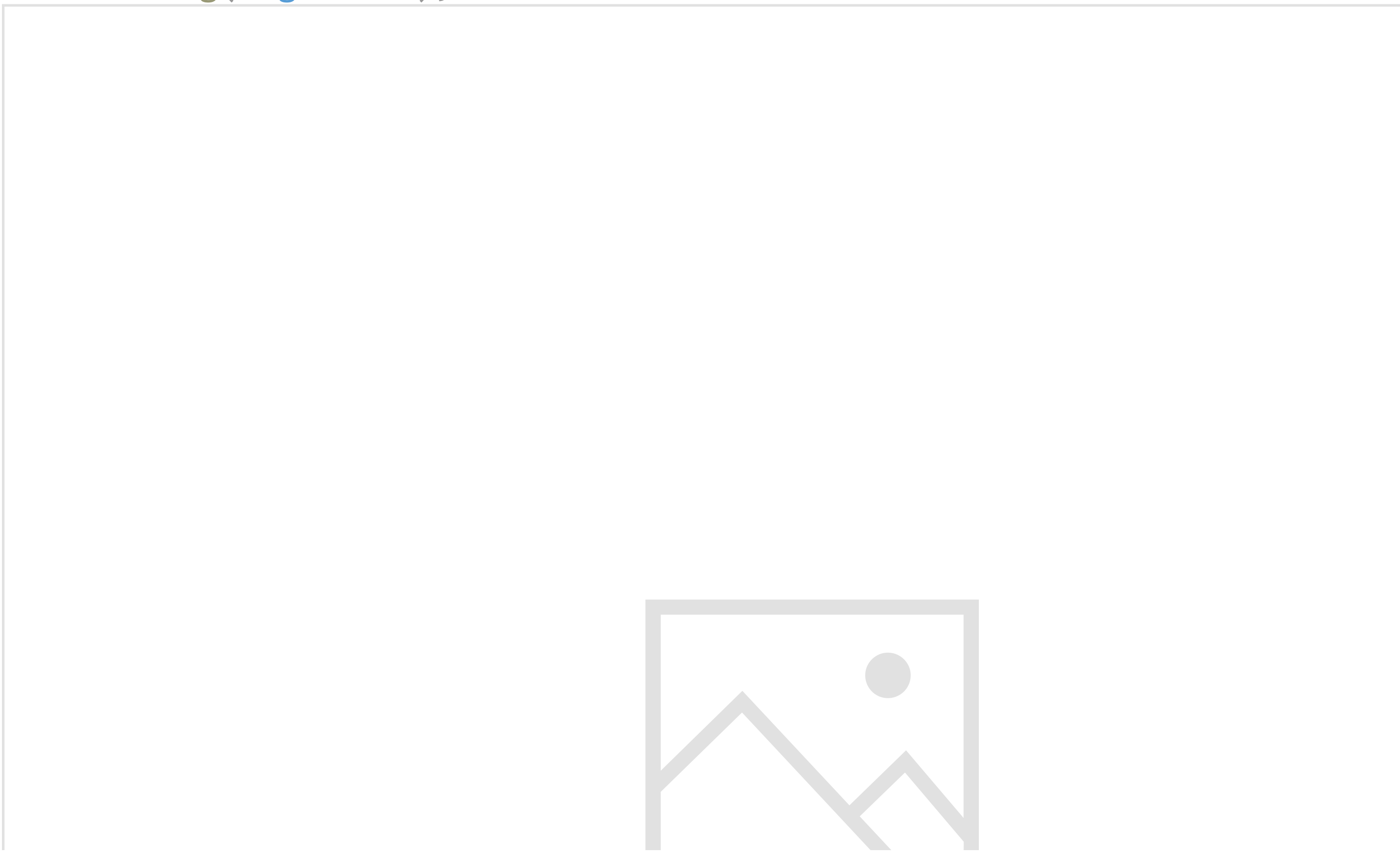
-Use exports to export multiple named variables
(exports.add = (a, b) => a + b);

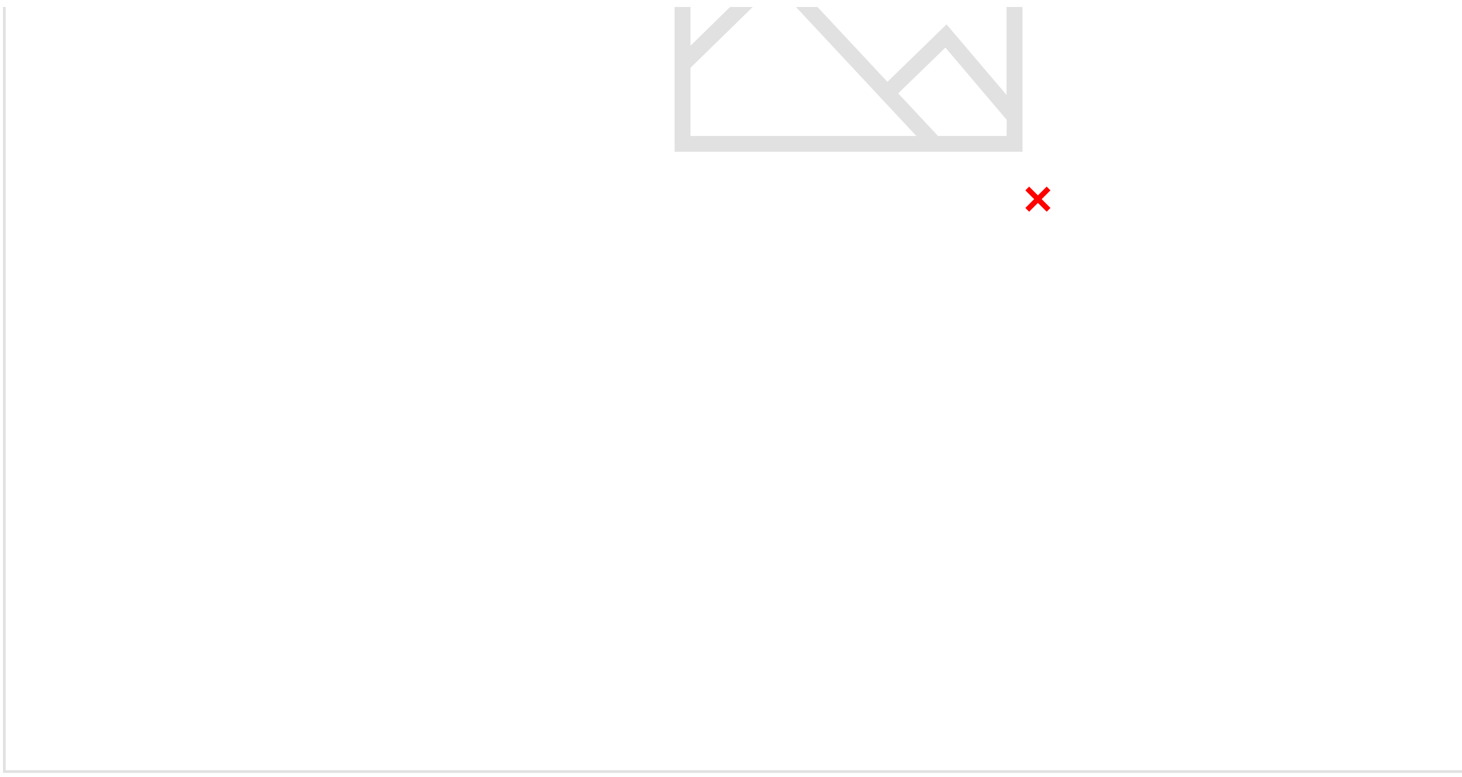
-This is how we import data from 1 module into another;

5. Caching:
Required modules are only executed at the 1st call => Cached to memory & wait to be called again

====**Requiring Modules in Practice**
modules.js:

```
// arguments = array in JS are objs we pass into a func
console.log(arguments);
```





0: exports
empty cuz we have NOT exported any modules yet

1: require()
2: module

3: __filename
'C:\\Users\\IGS\\Desktop\\Web\\vscode\\complete-node-bootcamp\\2-how-node-works\\starter\\modules.js',

4: __dirname
'C:\\Users\\IGS\\Desktop\\Web\\vscode\\complete-node-bootcamp\\2-how-node-works\\starter'

wrapper function:

```
console.log(require('module').wrapper);
```

```
[
  '(function (exports, require, module, __filename, __dirname) { ',
  '\n});'
]
```

Exporting a class variable:

test-module-1.js:

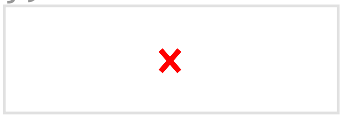
```
class Calculator {
    add(a, b) {
        return a+b
    }
    multiply(a, b) {
        return a*b
    }
    devide(a,b) {
        return a/b
    }
    remainder(a,b) {
        return a%b
    }
}
module.exports = Calculator;
```

modules.js:

```
const C = require('./test-module-1');
const calc1 = new C();
console.log(calc1.add(2, 5));
```

test-module-1.js:

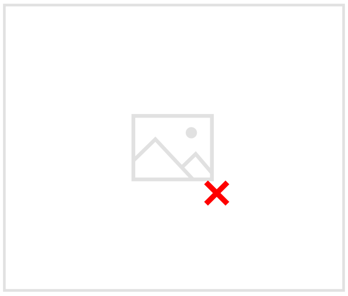
```
// Anonymous module.exports
module.exports = class {
    add(a, b) {
        return a+b
    }
    multiply(a, b) {
        return a*b
    }
    devide(a,b) {
        return a/b
    }
    remainder(a,b) {
        return a%b
    }
};
```



==== **When to use 'exports '**

test-module-2.js:
// exports.xx = (argu1, argu2) => {...}

```
exports.add = (a, b) => a+b;
exports.multiply = (a, b) => a*b;
exports.divide = (a, b) => a/b;
exports.remainder = (a, b) => a%b;
```

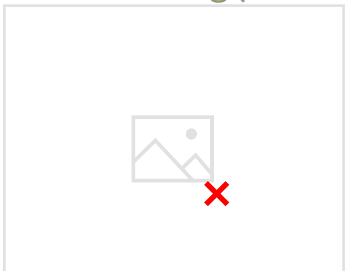


// ES6 destructuring:

```
// exports - ES6 destructuring
const { add, multiply, divide, remainder } = require('./test-
module-2');
console.log(add(2, 5));
console.log(multiply(2, 5));
console.log(divide(2, 5));
console.log(remainder(2, 5));
```

modules.js:

```
// exports
const calc2 = require('./test-module-2');
console.log(calc2.add(2, 5));
console.log(calc2.multiply(2, 5));
console.log(calc2.divide(2, 5));
console.log(calc2.remainder(2, 5));
```



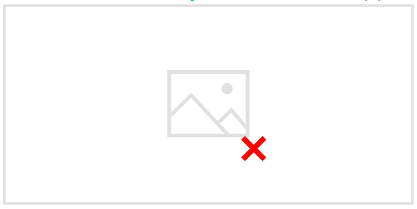
// Caching

modules.js:

```
// Caching
require('./test-module-3')();
```

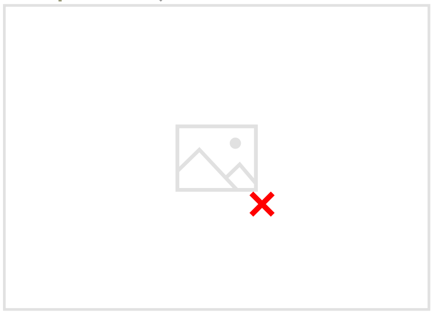
test-module-3.js:

```
console.log('Hello from the module');
module.exports = () => console.log('Log this text');
```



modules.js:

```
require('./test-module-3')();
require('./test-module-3')();
require('./test-module-3')();
```



// Technically, test-module-3.js only loaded once

// module.exports has exported the function to be cached & called