What is event loop? How does it work?

Why do we use Express.js?

**Difference between CommonJS modules and ES6 modules.**

1. We need explicitly specify file extensions in imports with ESM, while file extensions are totally optional with the CommonJS require function.
2. ES modules run implicitly in strict mode. This means that we don't have to explicitly add the "use strict" statements at the beginning of every file. Strict mode cannot be disabled; therefore, we cannot use undeclared variables or the ‘with’ statement or have other features that are only available in non-strict mode, but this is definitely a good thing, as strict mode is a safer execution mode.
3. In ESM, some important CommonJS references are not defined. These include *require, exports, module.exports, \_\_filename*, and *\_\_dirname*. If we try to use any of them within an ES module, since it also runs in strict mode, we will get a ReferenceError

In ESM, it is possible to get a reference to the current file URL by using the special object import.meta. Specifically, import.meta.url is a reference to the current module file in a format similar to file:///path/to/current\_module.js. This value can be used to reconstruct \_\_filename and \_\_dirname in the form of absolute paths:

import { fileURLToPath } from 'url'

import { dirname } from 'path'

const \_\_filename = fileURLToPath(import.meta.url)

const \_\_dirname = dirname(\_\_filename)

1. In the global scope of an ES module, ‘this’ is undefined, while in CommonJS, ‘this’ is a reference to exports
2. ESM cannot import JSON files directly as modules, a feature that is used quite frequently with CommonJS. The following import statement will fail:

import data from './data.json'

It will produce a TypeError (Unknown file extension: .json). To overcome this limitation, we can use again the module.createRequire utility:

import { createRequire } from 'module'

const require = createRequire(import.meta.url)

const data = require('./data.json')

console.log(data)