Concepte avansate in programarea pe Internet

Alex Ghiurău

Tehnologii WEB

REACT

Alex Ghiurău

You learn to speak by speaking,
to study by studying,
to run by running,
to work by working;
and just so, you learn to love by loving.

All those who think to learn in any other way deceive themselves.

– Francis de Sales

JavaScript Modules

"Write code that is easy to delete, not easy to extend."

Agenda

- Packages
- ES Modules
- Assignment

What is a Module?

Modules are an integral piece of any robust application's architecture and typically help in keeping the units of code for a project both cleanly separated and organised.

Packages

What is a Package?

A package is a chunk of code that can be distributed (copied and installed). It may contain one or more modules and has information about which other packages it depends on.

NPM node package manager

NPM is a repository of JavaScript packages.

Installing Node.js - https://nodejs.org/en/download/

$ node -v
$ npm -v

Installing Packages in Local Mode

$ mkdir project && cd project
$ npm init -y
$ npm install --save lodash

Uninstalling Packages in Local Mode

$ npm uninstall lodash

Re-installing Project Dependencies

$ npm install

ES Modules

ES Modules

- Static structure - It means that you can determine imports and exports at compile time (statically) – you only need to look at the source code, you don’t have to execute it.
- You can only import and export at the top level (never nested inside a conditional statement)
- import and export statements have no dynamic parts (no variables etc. are allowed)
- ES Module is file-based, and module instances are singletons

ES Modules

- ES Module uses “**import**” keyword to get dependencies and **“export”** keyword to expose something on the public API of the module.

```
import { maxBy } from 'lodash'

export const add = (a, b) => a + b;
export const subtract = (a, b) => a - b;
export const multiply = (a, b) => a * b;
export const divide = (a, b) => a / b;
export const max =  arr => maxBy(arr);

export default add;
```

Execution Order

ES modules act different as we would expect. The root cause is that **import** is not a function, but a **language keyword**. With language keywords the compiler can traverse and parse the whole application tree before executing it. Only a**fter traversing** does the compiler start executing them **from bottom to top**.

```
// index.js
console.log('loading module');
import './module';
console.log('module loaded');

// module.js
console.log('hello from module');
```

```
// hello from module
// loading module
// module loaded
```

Why use Modules?

Why use?

- **Maintainability –** Since modules are self-contained, they reduce the dependencies on parts of the codebase as much as possible to improve them independently.

Also, updating a single module is much more convenient than updating the complete code. Updating one module doesn’t affect the other modules.

Why use?

- **Namespacing –** Since variables outside the scope of a top-level function are global, it's common to have "namespace pollution" in our code. This means that totally unrelated code shares those global variables, creating confusion, resulting in unexpected outputs.

Modules help us avoid these situations by creating private space for the variables.

Why use?

- **Reusability –** JavaScript modules let us copy the previous code into our program and reuse it over and over again.

Writing the complete code for the same task is time-consuming; modules provide us with a better approach to optimize our code.

References for homework

(to read)

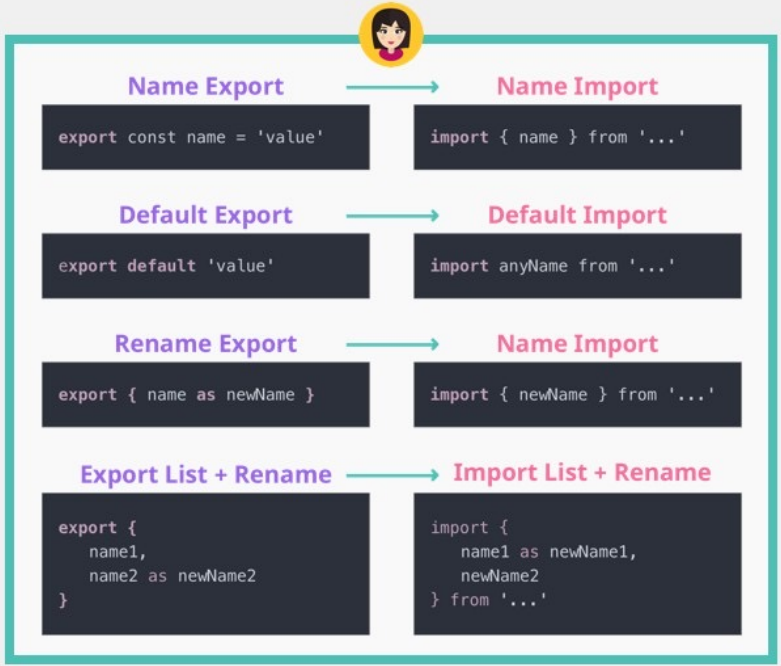
References

- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Modules
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/export
- https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Statements/import
- https://exploringjs.com/es6/ch_modules.html
- https://2ality.com/2014/09/es6-modules-final.html#more-on-importing-and-exporting

JavaScript

Module

Cheatsheet



samantha_ming samanthaming.com samanthaming

Assignment

Analog clock - Assignment

Generate an analog clock using HTML, CSS and Javascript that has the following characterstics

Analog clock - Assignment

- Displays only 12, 3, 6, 9 as hours (as readable numbers)
- For all other hours it should display only a line
- For minutes a smaller line
- Hours hand is thicker
- Minutes hand is thinner

Analog clock - Assignment

- It should have a (stopwatch) button that does the following:
  - Sets a timer (e.g. 1 minute, 30 seconds, 2 minutes)
  - This should reset the hour and minutes hands to 12’ o clock
  - Spins only the seconds hand for as long as the stopwatch timer is set

Have patience with all things, but chiefly have patience with yourself.

Do not lose courage in considering your imperfections
but instantly set about remedying them – every day begin the task anew.”.

– Francis de Sales