Práctica Módulo 8 - Trabajando con imports

Introducción

En la práctica de este módulo vamos a mostrar en un documento HTML una lista de usuarios.

Un usuario vendrá representado con la siguiente estructura de datos:

```
{
  "id": 1,
  "avatar": "https://robohash.org/voluptatemdelenitinostrum.jpg?
size=50x50&set=set1",
  "first_name": "Willem",
  "last_name": "Waterfall",
  "email": "wwaterfall0@t-online.de"
}
```

Crearemos funciones sencillas para realizar operaciones sobre estos datos y gestionar los elementos HTML donde representarlos:

- getUsers: devolverá el array completo de usuarios.
- getFullName: devolverá el nombre completo del usuario, concatenando los atributos *first_name* y *last_name*.
- getAvatar: devolverá un elemento *img* de HTML con la imagen del atributo *avatar*.
- getUserNode: devolverá la siguiente estructura de HTML con la información del usuario:

```
<div>
  <img src="https://robohash.org/voluptatemdelenitinostrum.jpg?
size=50x50&set=set1" /> Willem Waterfall - wwaterfall0@t-online.de
</div>
```

Separaremos la funcionalidad en tres módulos: *data-business* (con la función *getUsers*), *avatar-business* (con *getAvatar*) y *user-business* (con *getFullName* y *getUserNode*).

Para la realización de la práctica, pasaremos por las diferentes formas de organizar los módulos, desde utilizar los imports con las etiquetas *script* en el propio documento, hasta utilizar el sistema de módulos de ES6, pasando por los de CommonJS.

> Nota: Para obtener el array de usuarios, utilizaremos un servicio de generación de mocks (datos falsos) con la estructura que nos interesa: https://mockaroo.com/

Ejercicio 1 - Importar con etiquetas *script* desde HTML

Crear módulo de datos

Vamos a crear un fichero *data-business.js* con la función *getUsers* que nos devolverá el array de datos que hemos creado con el generador de mocks. En este documento el array sólo contendrá un objeto para mantener la legibilidad:

data-business.js

```
function getUsers() {
  return [
    {
      id: 1,
      avatar:
        "https://robohash.org/voluptatemdelenitinostrum.jpg?
size=50x50&set=set1",
      first_name: "Willem",
      last_name: "Waterfall",
      email: "wwaterfall0@t-online.de"
    },
  ];
```

Crear módulo de avatar

Para crear el módulo de avatar, añadiremos un nuevo fichero *avatar-business.js* con la función *getAvatar*:

avatar-business.js

```
function getAvatar(user) {
  const img = document.createElement("img");
  img.src = user.avatar;

  return img;
}
```

Crear módulo de usuario

El módulo de usuarios lo crearemos sobre el fichero *user-business.js* con las funciones que gestionan los datos del usuario:

user-business.js

```
function getFullName(user) {
  return user.first_name + " " + user.last_name;
}

function getUserNode(user) {
  const node = document.createElement("div");
  node.appendChild(getAvatar(user));
  node.append(getFullName(user) + " - " + user.email);
```

```
  return node;
}
```

Crear el fichero index de entrada

Para orquestar las llamadas a las funciones anteriores, crearemos un fichero *index.js*, desde el que obtendremos el listado de usuarios y crearemos los nodos con los datos de cada uno para mostrarlos en nuestro documento HTML.

index.js

```
const users = getUsers();
const nodes = [];
for (let user of users) {
  const node = getUserNode(user);
  nodes.push(node);
}

window.onload = function() {
  for (let node of nodes) {
    document.getElementById("root").append(node);
  }
};
```

Documento HTML

Por último crearemos el documento HTML donde mostrar el listado de usuarios, importando los scripts anteriores en el orden correcto:

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Práctica Módulo 8</title>
    <script src="./data-business.js"></script>
    <script src="./avatar-business.js"></script>
    <script src="./user-business.js"></script>
    <script src="./index.js"></script>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

Ahora corriendo la aplicación con el comando *lite-server* podremos ver el listado de usuarios en nuestro documento.

> Nota: jugar a cambiar el orden de los imports desde HTML para ver los distintos errores que se muestran por consola.

Ejercicio 2 - Utilizar el objeto *window* para organizar la funcionalidad

Partiendo del ejercicio anterior, utilizaremos funciones autoinvocadas para crear en el objeto *window* de nuestro documento las distintas funcionalidades que necesitamos:

datos-business.js

```
(function(App) {
  App.getUsers = function() {
    return [
      {
        id: 1,
        avatar:
          "https://robohash.org/voluptatemdelenitinostrum.jpg?
size=50x50&set=set1",
        first_name: "Willem",
        last_name: "Waterfall",
        email: "wwaterfall0@t-online.de"
      }
    ];
  };
})(window.App || (window.App = {}));
```

avatar-business.js

```
(function(App) {
  App.getAvatar = function(user) {
    const img = document.createElement("img");
    img.src = user.avatar;

    return img;
  };
})(window.App || (window.App = {}));
```

user-business.js

```
(function(App) {
  App.getFullName = function(user) {
    return user.first_name + " " + user.last_name;
  };
```

```
  App.getUserNode = function(user) {
    const node = document.createElement("div");
    node.appendChild(App.getAvatar(user));
    node.append(App.getFullName(user) + " - " + user.email);

    return node;
  };
})(window.App || (window.App = {}));
```

index.js

```
(function(App) {
  const users = App.getUsers();
  const nodes = [];
  for (let user of users) {
    const node = App.getUserNode(user);
    nodes.push(node);
  }

  window.onload = function() {
    for (let node of nodes) {
      document.getElementById("root").append(node);
    }
  };
})(window.App || (window.App = {}));
```

index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Práctica Módulo 8</title>
    <script src="./data-business.js"></script>
    <script src="./avatar-business.js"></script>
    <script src="./user-business.js"></script>
    <script src="./index.js"></script>
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>
```

Ejercicio 3 - Sistema de módulos de CommonJS

> Para este ejercicio, al necesitar la librería de CommonJS, vamos a utilizar CodeSandbox para crear nuestra aplicación. El ejemplo completo lo tenéis disponible en https://codesandbox.io/s/bootcamp-lemoncode-mod8-practica-3-12jfj

Ejercicio 4 - Sistema de módulos de ES6

> Como en el ejercicio anterior, utilizaremos un CodeSandbox para crear nuestra aplicación. El ejemplo completo está disponible en https://codesandbox.io/s/bootcamp-lemoncode-mod8-practica-4-yn34k