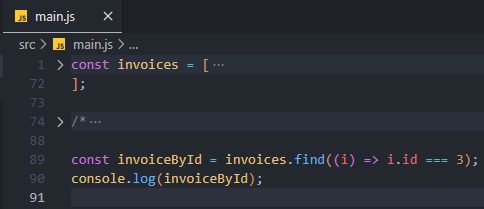
Arreglos y métodos find, filter y otros

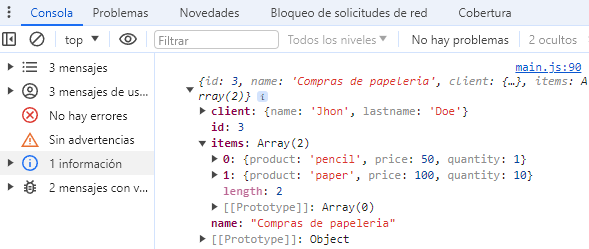
En esta sección vamos a aplicar distintos métodos en un arreglo que contiene una lista de objetos. Es importante reconocer que JavaScript proporciona una variedad de métodos incorporados que son útiles para manipular objetos y arreglos.

Ten en cuenta que se está utilizando el mismo objeto **invoices** de la sección anterior y los métodos que no se están utilizando se muestran en forma de comentario.

El método find

El método **find** se utiliza para encontrar el primer elemento en un arreglo que cumpla con una condición determinada. Esta condición se especifica mediante una función de prueba que devuelve un valor booleano (si es **true**, devuelve ese elemento, de lo contrario devuelve un **undefined**).

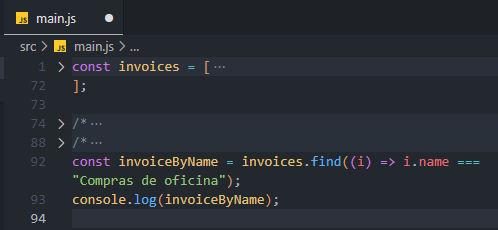


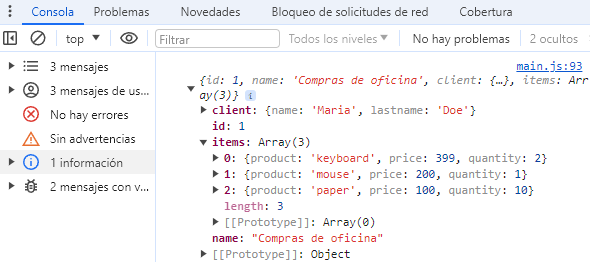


El resultado del ejemplo mostrado es el primer objeto con toda la información de la factura que tiene el **id** igual a 3.

Buscar un objeto por un valor especificado en su atributo

El método **find** también puede ser utilizado para buscar un objeto en un arreglo basado en el valor de uno de sus atributos.

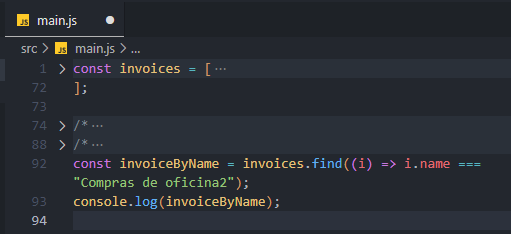


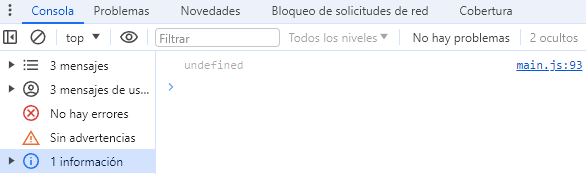


El resultado del ejemplo mostrado es el primer objeto con toda la información de la factura que tiene un atributo **name** igual a **"Compras de oficina"**.

Buscar un objeto que no existe

Como ya se había mencionado antes, cuando se utiliza el método **find**, si la condición de búsqueda no se cumple para ningún objeto en el arreglo, el método devuelve **undefined**.

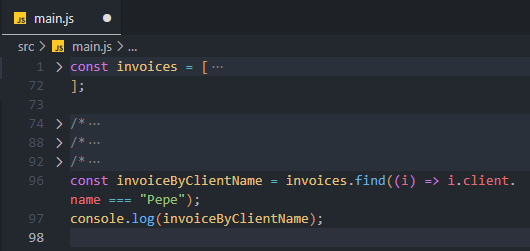


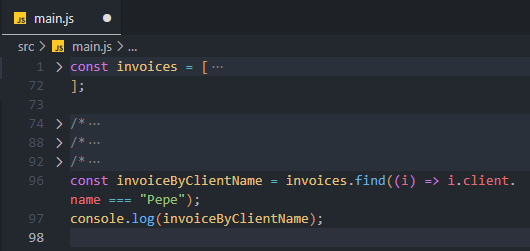


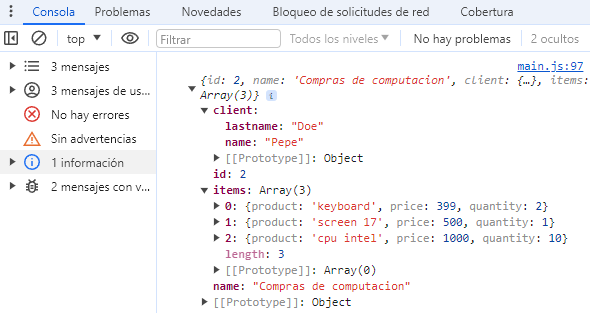
El resultado del ejemplo mostrado es el primer objeto con toda la información de la factura que tiene un atributo **name** igual a **"Compras de oficina2"**. Como dicho objeto no existe en el arreglo, se devuelve un **undefined**.

Buscar un objeto por un valor especificado en un atributo del objeto anidado

El método **find** también puede utilizarse para buscar un objeto en un arreglo basado en un valor específico de un atributo dentro de un objeto anidado.





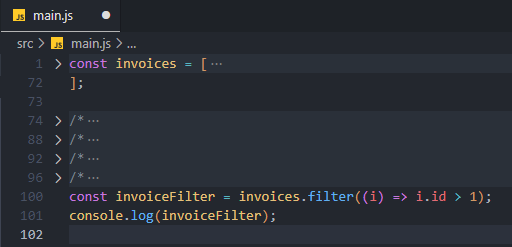


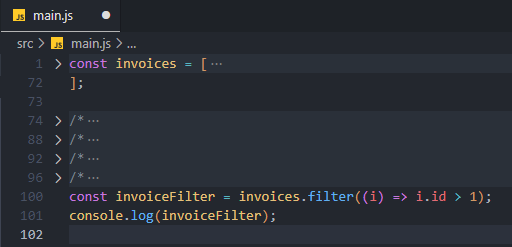
El resultado del ejemplo mostrado es el primer objeto con que tenga un atributo **client.name** igual a **"Pepe"**.

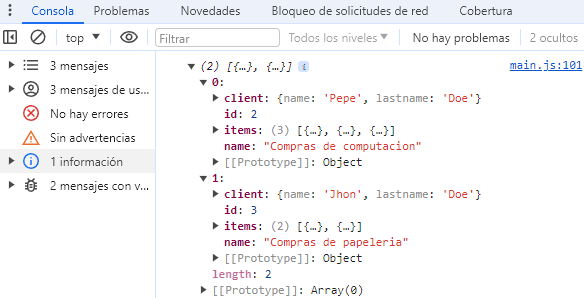
Es importante recordar que, si una función de flecha ocupa una sola línea de código, se puede eliminar el **return** y las llaves.

El método filter

El método **filter** se utiliza para crear un nuevo arreglo con todos los elementos que cumplan una condición determinada. A diferencia del método **find**, que devuelve el primer elemento que cumpla con la condición, **filter** devuelve todos los elementos que la cumplan.





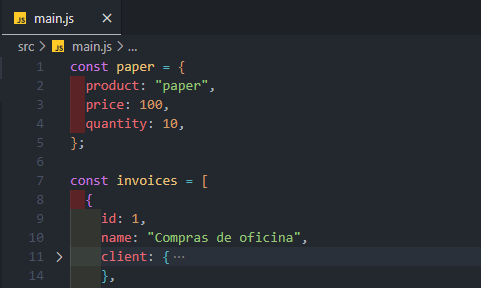


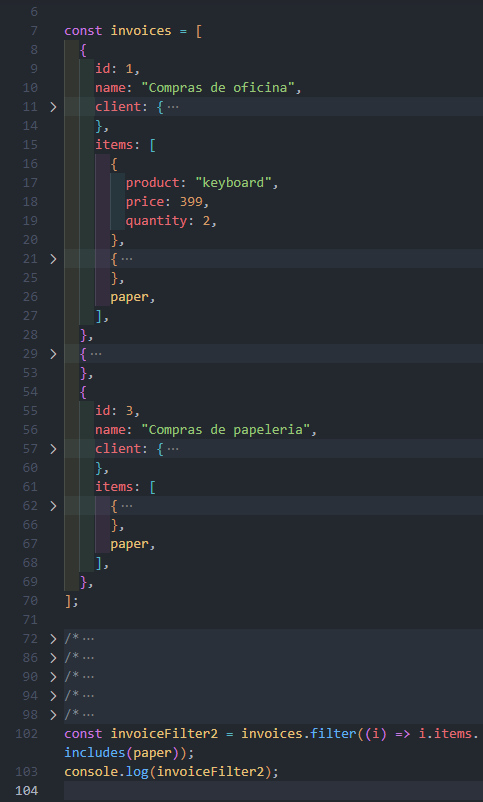
El resultado del ejemplo mostrado es un nuevo arreglo que contiene solo los objetos cuyo atributo **id** es mayor que **1**.

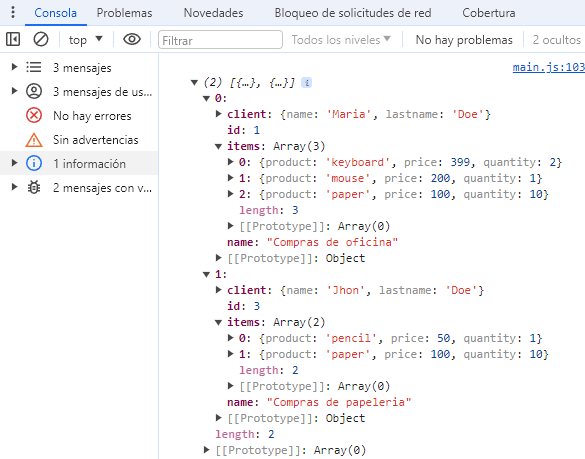
El método includes

El método **includes** se utiliza para determinar si un arreglo incluye un determinado valor, devolviendo **true** o **false** según corresponda. Es importante destacar que realiza una comparación por el tipo de datos.

Para el siguiente ejemplo se define un nuevo objeto denominado **paper**, el cual va a hacer referencia a un producto de la lista de ítems. Luego se instancia el objeto **paper** al arreglo de **items** definido en el objeto **invoice**. Dicho de otra forma, reemplaza con la constante **paper** el elemento de la lista **items** que contiene como valor **"paper"** en el atributo **product**.



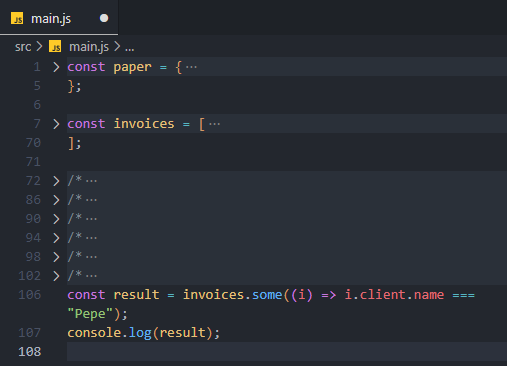


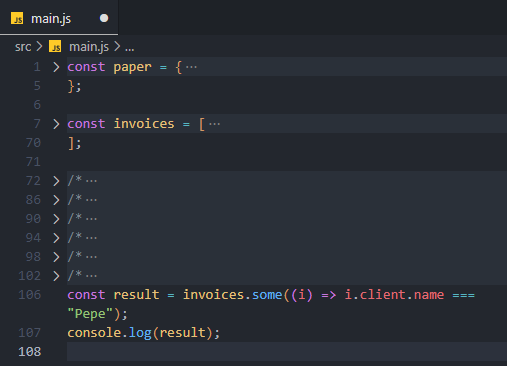


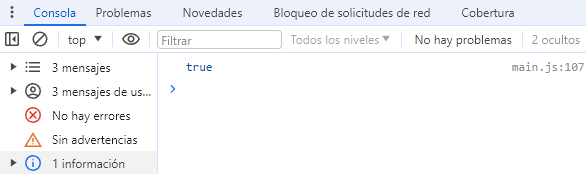
En este ejemplo, se utiliza el método **filter** en el arreglo **invoices** para obtener un nuevo arreglo de facturas. La condición dentro de **filter** verifica si la lista de ítems de cada factura (**i.items**) incluye el objeto **paper**, este último se define con el método **includes**. Si es así, esa factura se incluye en el nuevo arreglo **invoiceFilter2**.

El método some

El método **some** se utiliza para comprobar si al menos uno de los elementos de un arreglo cumple con una condición determinada. Devuelve **true** si se encuentra al menos un elemento que cumple con la condición, y **false** en caso contrario.



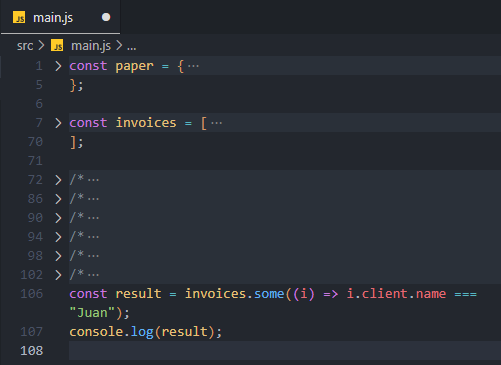


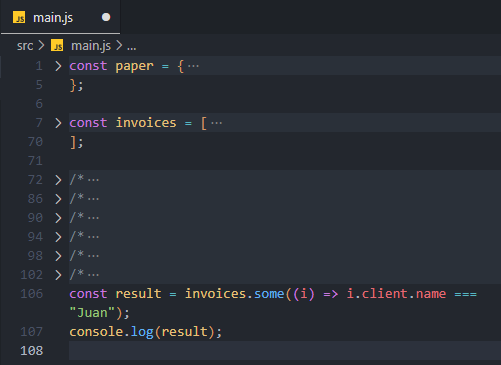


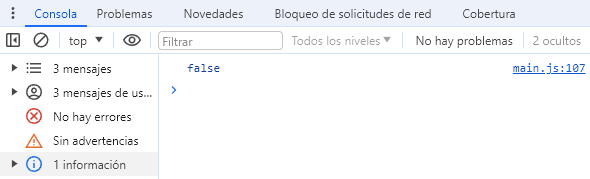
En el ejemplo mostrado, el método **some** se utiliza para comprobar si hay al menos un objeto en el arreglo **invoices** cuyo atributo client.name es igual a **"Pepe"**.

Comprobar que no exista un elemento

Como ya se ha mencionado antes, el método **some** puede devolver un **false** en el caso de que no existiera al menos un elemento del arreglo que cumpla con la condición determinada.



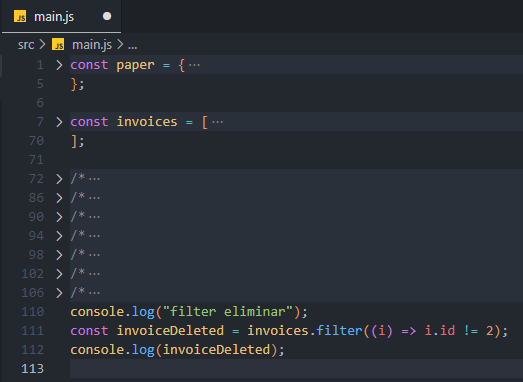


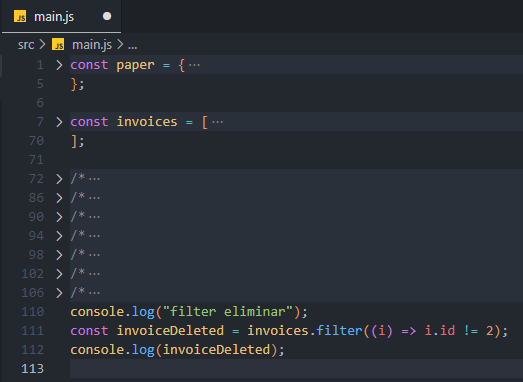


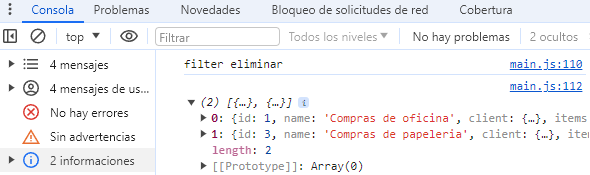
En el ejemplo mostrado, al intentar encontrar un cliente con el nombre **"Juan"**, la variable **result** será **false**, ya que ninguno de los objetos en el arreglo **invoices** tiene el atributo **client.name** igual a **"Juan"**.

Operadores de diferenciación en el método filter

El método **filter** puede utilizarse para eliminar un elemento de un arreglo de forma inmutable, lo que significa que no modifica el arreglo original. Para lograr esto, se puede usar un filtro basado en el arreglo original y crear un nuevo arreglo que excluya el elemento que se desea eliminar. Esto se puede lograr utilizando los operadores de diferenciación **!=** o **!==**.



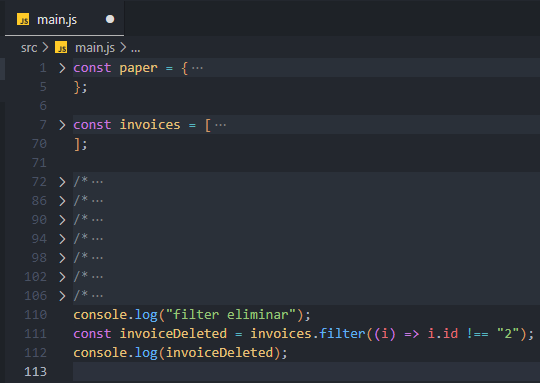


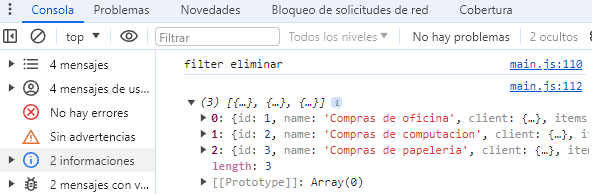


En el ejemplo mostrado, el objeto con **id** igual a **2** es eliminado del arreglo **invoices** y el nuevo arreglo **invoiceDeleted** contiene los objetos restantes.

El operador no estrictamente igual (!==)

El operador !**==** comprueba la desigualdad entre dos valores, pero no realiza la conversión de tipos de datos antes de hacer la comparación.

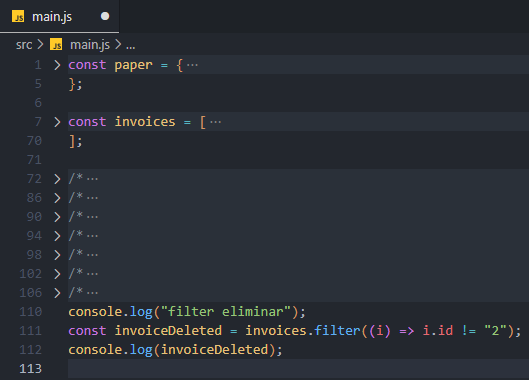


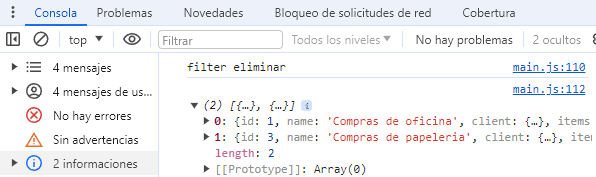


En el ejemplo mostrado, al usar **i.id !== "2"**, no se elimina el objeto porque se compara un número con una cadena de texto, que no son idénticos en tipo de datos.

El operador no igual (!=)

El operador **!=** comprueba si los dos valores son desiguales, pero realiza la conversión de tipos de datos si es necesario antes de hacer la comparación.





En el siguiente mostrado, el objeto con el identificador **2** se elimina del arreglo **invoices** porque el operador **!=** compara solo los valores y convierte automáticamente los operandos al mismo tipo antes de la comparación.

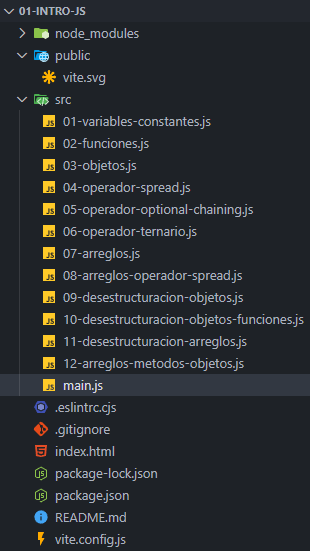
Antes de pasar al siguiente tema

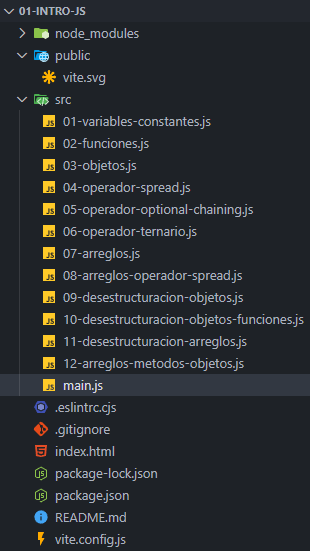
Realiza estas acciones en Visual Studio Code:

1. Abre el panel izquierdo y busca el archivo **main.js**.
2. Haz clic derecho en el archivo y selecciona **Cambiar nombre**.
3. Cambia el nombre del archivo a **12-arreglos-metodos-objetos.js** y presiona **Enter**.
4. Ahora, crea un nuevo archivo, dentro la carpeta raíz **src**, haciendo clic en el botón **Nuevo archivo** en la parte superior del panel izquierdo.



1. Nombre el nuevo archivo como **main.js** y presiona **Enter**.





A partir de este momento, se realizarán cambios en el archivo **main.js** que has creado.