Operador Optional Chaining

En esta sección abordaremos el manejo del operador Spread, se tiene en cuenta que el manejo de atributos no definidos y objetos no definidos es fundamental al trabajar con objetos en JavaScript, especialmente cuando se obtienen datos de fuentes externas como APIs REST.

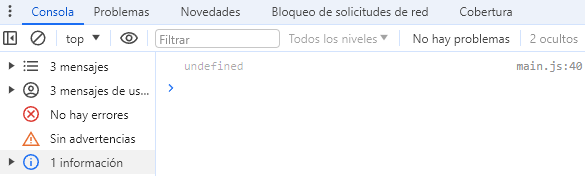
Problema con los atributos no definidos

Los atributos de un objeto pueden ser opcionales y podrían estar ausentes en ciertas circunstancias, como cuando se obtienen de un servicio remoto. Si intentamos acceder a un atributo que no está definido, JavaScript simplemente devolverá **undefined**.

Ten en cuenta que el objeto **invoice** es el mismo de la sección anterior.

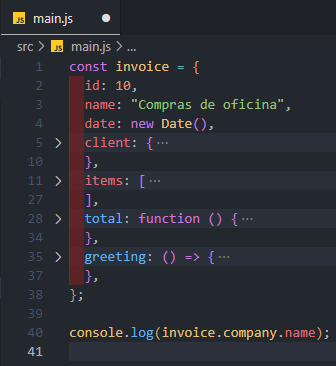


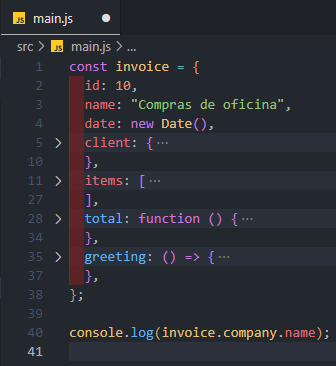


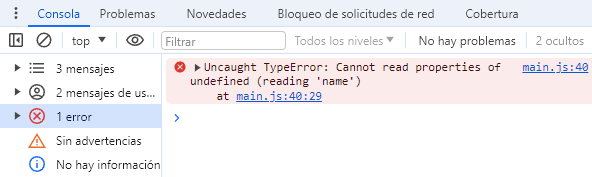


Problema con los objetos no definidos

Si intentamos acceder a un atributo de un objeto que no está definido, JavaScript lanzará un error fatal, interrumpiendo la ejecución del programa.



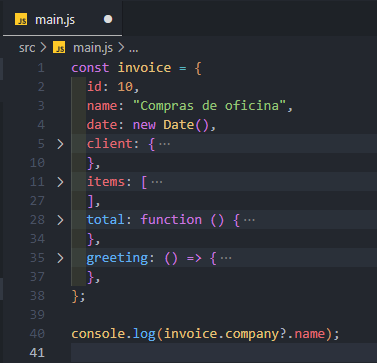


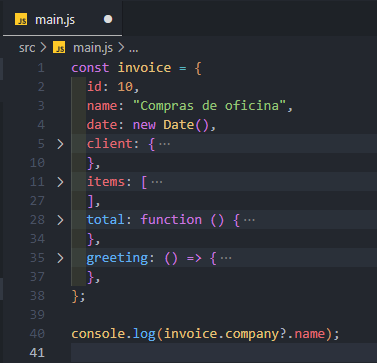


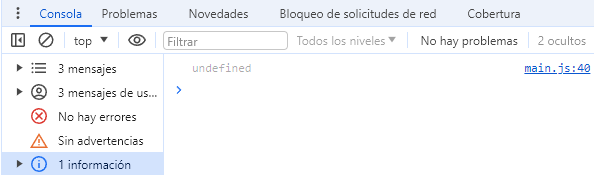
El error en la consola indica que no se puede leer las propiedades (atributo y valor) indefinidas, en este caso, el atributo **name**.

El operador Optional Chaining

El operador Optional Chaining (**?.**) es una característica introducida en ECMAScript 2020 que nos permite acceder al valor de una propiedad dentro de una cadena de objetos sin tener que comprobar explícitamente si cada referencia en la cadena es válida.





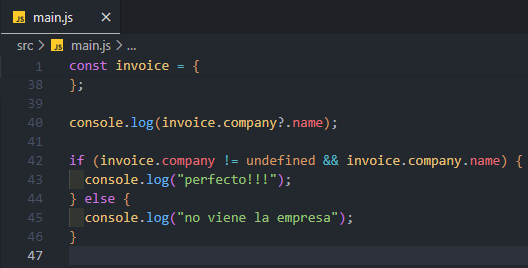


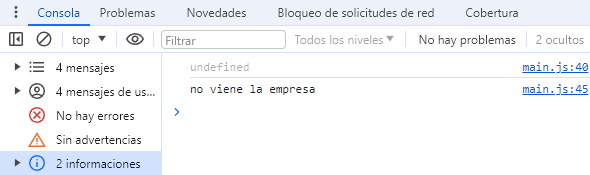
En el ejemplo mostrado, al utilizar **invoice.company?.name**, si **invoice.company** es **undefined** o **null**, la expresión se evaluará a **undefined** y no se lanzará ningún error. Esto proporciona una forma más concisa y segura de acceder a propiedades anidadas en objetos.

Antes de la introducción del operador

Antes de la introducción del operador Optional Chaining, la verificación de propiedades anidadas implicaba el uso de condicionales para asegurar la existencia de cada propiedad en la cadena de objetos.

Esto aún es una práctica válida y se utilizaba para evitar errores cuando alguna de las propiedades en la cadena es **null** o **undefined**.





Casos de uso del Operador Optional Chaining

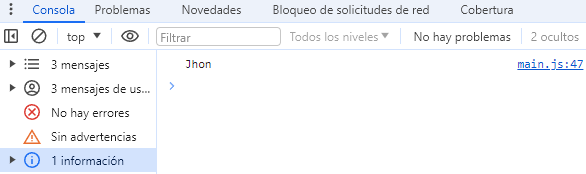
Es importante tener en cuenta los siguientes casos en donde se aplica el Operador Optional Chaining en objetos y atributos.

Caso 1: Evaluar objetos y atributos que existen

En este caso, el operador Optional Chaining evalúa si el objeto anidado **client** y el atributo **name** existen en el objeto **invoice**.

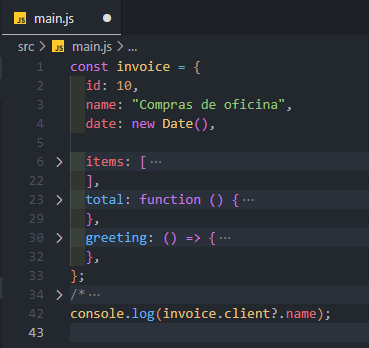


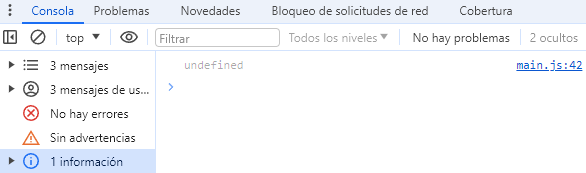




Caso 2: Evaluar objetos y atributos que no existen

Aquí se muestra cómo el operador Optional Chaining permite acceder a propiedades que pueden no estar definidas sin provocar errores en la ejecución del código.

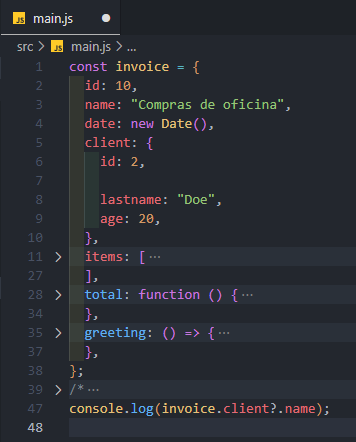


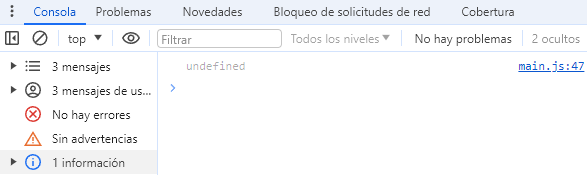


En el ejemplo mostrado, se muestra un **undefined** en la consola porque el objeto anidado **client** y sus atributos fueron eliminados en el código.

Caso 3: Evaluar objetos que existen, pero con sus atributos que no existen

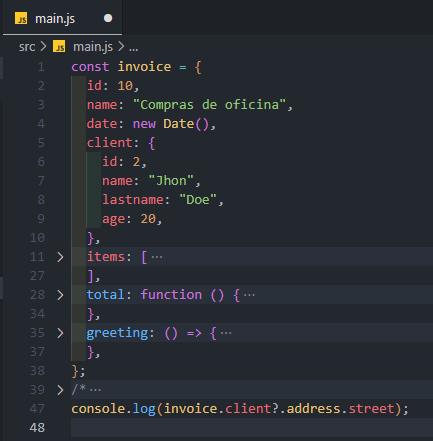
En este caso, el operador Optional Chaining maneja la ausencia del atributo **name** en el objeto anidado **client**.

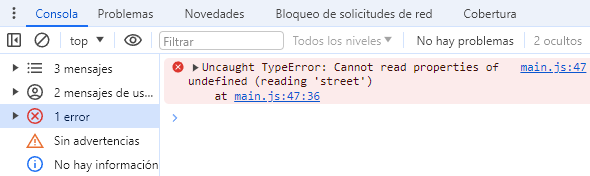




El problema con los objetos anidados que no existen

Al intentar acceder a propiedades de objetos anidados que no existen, se genera un error grave en JavaScript. Esto ocurre debido a que el intento de acceder a una propiedad de un objeto **undefined** provoca un error.



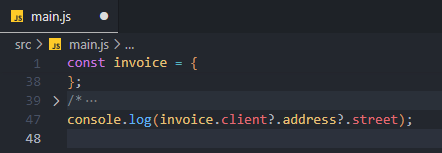


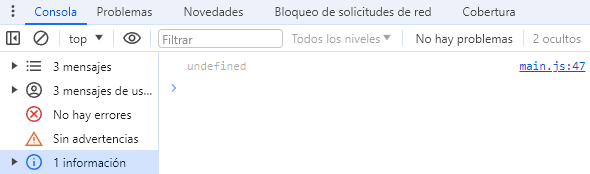
En el ejemplo mostrado, **address** podría ser un objeto anidado en el objeto **client**, el cual está anidado al objeto **invoice**.

Aplicando el operador Optional Chaining

Para evitar errores al acceder a propiedades de objetos anidados que pueden no existir, se utiliza el operador Optional Chaining (**?.**). Con este operador, el código se vuelve más robusto y menos propenso a errores al trabajar con objetos anidados.

Ten en cuenta que el operador Optional Chaining se utiliza en objetos anidados; en el siguiente ejemplo, en los objetos **client** y **address**.





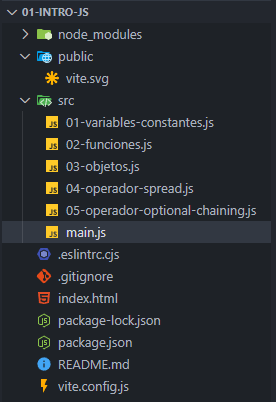
Antes de pasar al siguiente tema

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 **05-operador-optional-chaining.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.