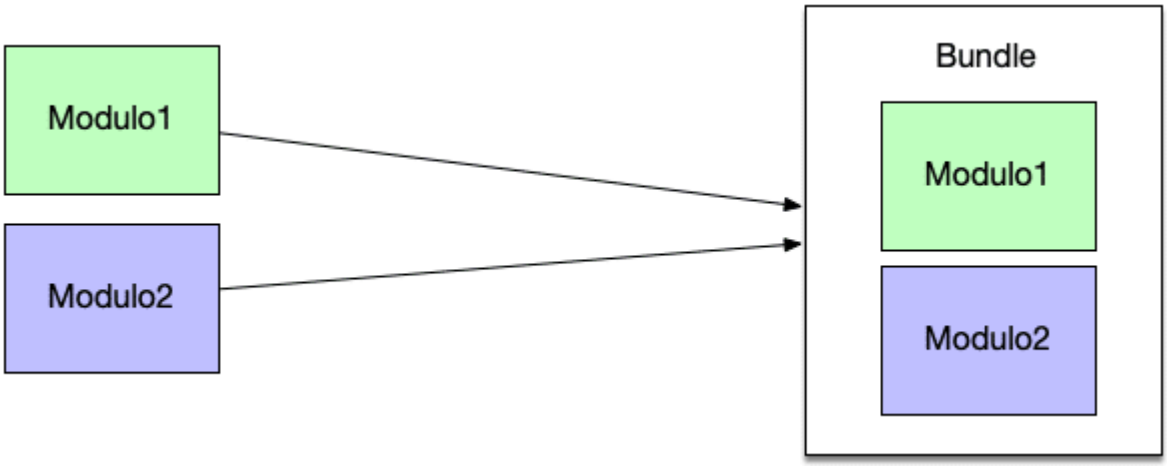
El concepto de JavaScript Module Browser es poco conocido , pero hoy en día los navegadores lo empiezan a soportar de una forma bastante sólida. En muchas ocasiones nos encontramos con bloques de código que tienden hacia la modularidad en JavaScript. Por ejemplo una clase como puede ser la clase de Matemáticas que hace calculos básicos y un programa que la use. Habitualmente se suele usar WebPack como JavaScript Bundle Para lo que tenemos programado como dos ficheros independientes fusionarlo y publicarlo en el navegador como un fichero que tenga todo integrado.



Es una opción más que razonable el problema es que haces tenemos que realizar alguna operación básica que implica modularidad y nos vemos obligados a usar este tipo de herramientas si o sí. A veces se nos olvida que el propio navegador ya soporta gestión de módulos de JavaScript en sus ultimas versiones y nos podemos ahorrar el uso de las herramientas. Vamos a verlo , para ello partiremos de la clase de Matématicas que suma y resta.

```
export class Matematicas {

    sumar(a,b) {

        return a+b;
    }
    restar(a, b) {
```

```
        return a-b;
    }
}
```

Esta clase como vemos se encuentra en un fichero independiente y deseamos usarla en otro programa que se encargue de importarla y sacar un resultado de la suma por la consola.

```
import { Matematicas } from "./matematicas.js";
let objeto=new Matematicas();
console.log(objeto.sumar(2,2));
```

JavaScript Module Browser

¿Como podemos conseguir que estos dos ficheros funcionen en un navegador de forma natural? . Pues muy sencillo el navegador soporta en las etiquetas de script el atributo module. Este atributo le permite gestionar módulos de JavaScript de forma natural.

```
<html>
    <head>
        <script  src="matematicas.js" type="module">
        </script>
            <script  src="001.js" type="module">
            </script>
    </head>
    <body>
    </body>
</html>
```

De esta manera si nosotros ejecutamos la pagina el navegador se encargará de cargar los módulos de JavaScript en su orden correcto y ejecutar el código del programa 001.js que suma dos números a través de la clase de Matemáticas.



Poco a poco los conceptos de modularidad van encajando también en los navegadores.

Otros artículos relacionados

- JavaScript Nested Destructuring
- JavaScript Map ES6 y el concepto de diccionario
- JavaScript Hoisting y sus trucos
- JavaScript ES6