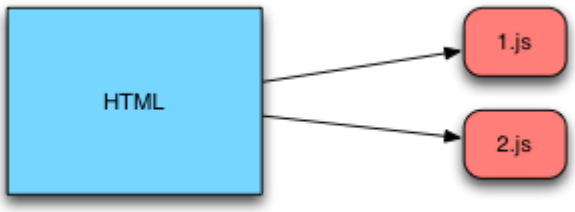
El concepto de JavaScript Bundle genera muchos problemas a los desarrolladores cuando diseñan arquitecturas JavaScript. Estamos acostumbrados a tener un código de JavaScript pequeño que se almacena en un par de ficheros y se carga en la correspondiente página web.



Lamentablemente esto no es lo que el futuro nos depara , cada día tenemos más y más frameworks de JavaScript . Esto implica que el código que escribimos de cliente cada día es mayor y el código de servidor se reduce. Ante este tipo de situaciones es imposible mantener toda nuestra lógica de cliente en un par de ficheros. Necesitamos organizarnos de una mejor manera.

JavaScript y Módulos

JavaScript define un sistema de módulos a través de Commons.js . Este sistema permite a un desarrollador definir los módulos que desee para su posterior reutilización. Vamos a crear un ejemplo concreto:

```
//suma.js
function suma(a,b) {

 return a+b;
}
```

```
exports.suma=suma;
```

```
//resta.js
function resta(a,b) {

return a-b;
}
exports.resta=resta;
```

Acabamos de definir un par de funciones, cada una es completamente independiente y se encuentra en fichero. Es comento de agruparlas usando commons.js . Para ello generamos un nuevo fichero “calculadora.js” que agrupa los dos módulos anteriores.

```
//calculadora.jps
var suma=require("./suma.js");
var resta=require("./resta.js");

exports.suma=suma.suma;
exports.resta=resta.resta;
```

Exportamos ambas funciones del módulo y las tendremos a nuestra disposición. Las deseamos ejecutar en un programa main:

```
//main.js
var operaciones=require("./calculadora.js")
```

```
console.log(operaciones.suma(2,2));
```

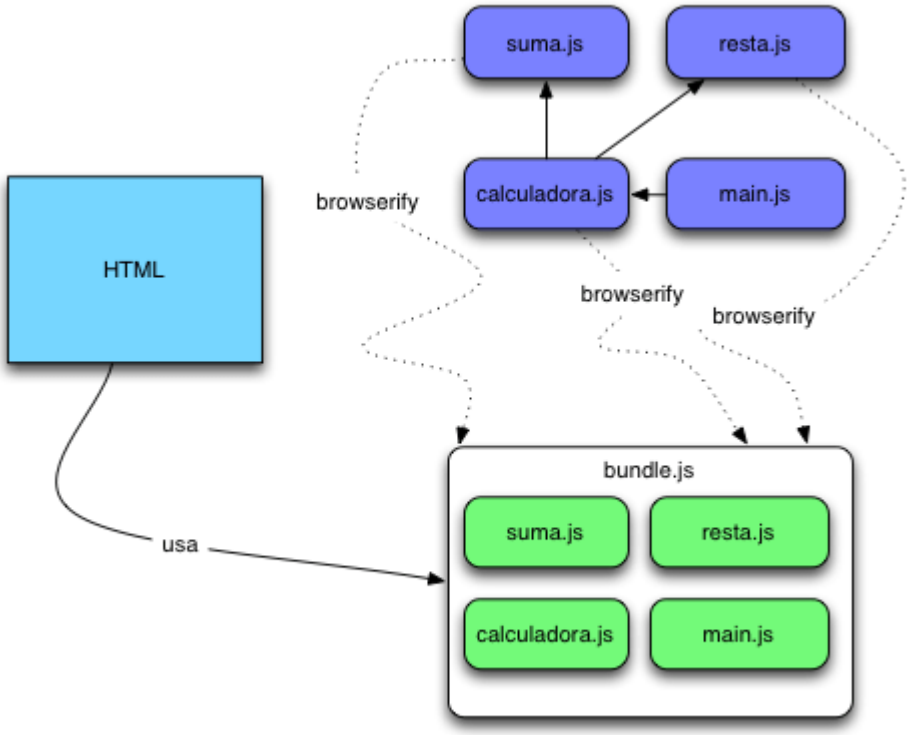
El resultado lo veremos impreso en la consola

```
iMac-de-cecilio:commons.js cecilio$ node main.js
4
iMac-de-cecilio:commons.js cecilio$
```

Todo funciona pero tenemos un problema importante. Este código funciona de forma perfecta en el servidor. Sin embargo no ocurriría lo mismo en un browser ya que este tendría que bajarse cada uno de los 4 ficheros de JavaScript y no tiene un soporte de commons. Esto genera problemas.

JavaScript Bundle y Browserify

Para solventar el problema de manejar tantos ficheros js podemos utilizar Browserify y construir un JavaScript Bundle. ¿Para qué sirve un Bundle?.



Un bundle sirve para agrupar todos los ficheros de JavaScript en uno solo. Así el navegador no necesita hacer varias peticiones HTTP. Vamos a instalar browserify usando npm.

npm install -g browserify

Este nos lo instalará en nuestro equipo a nivel global . El siguiente paso es ejecutarlo contra el fichero main.js y asignar el nombre del fichero de destino.

browserify main.js> bundle.js

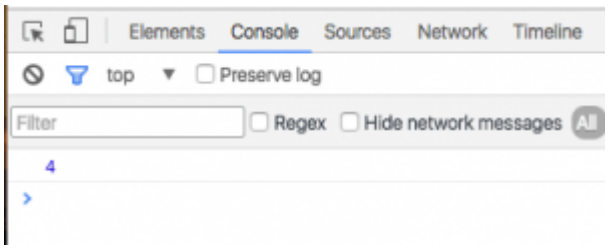
Browserify genera un nuevo fichero que incluye toda la funcionalidad que estaba ubicada en los demás. Usamos una página web sencilla para ejecutarlo.

```
<html>
<head>
<script type="text/javascript" src="bundle.js">
</script>

</head>
<body>

</body>
</html>
```

Cargamos la página en un navegador e imprimirá 4 por pantalla .



Acabamos de construir nuestro primer bundle.

Otros artículos relacionados: JavaScript Streams , RxJs y la programación reactiva