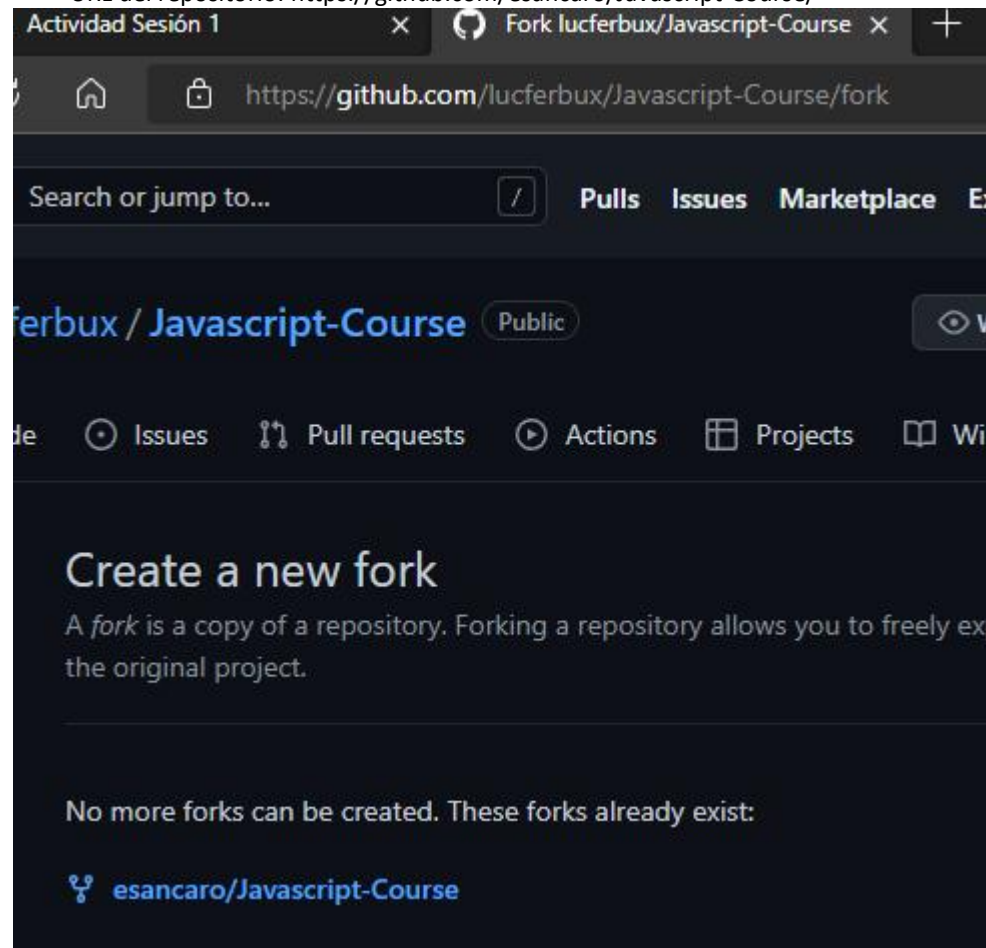


# Actividad Sesión 1

Alumno: Esteban Antonio Castro Rojas  
Fecha: 2022-06-28

## Ejercicio 1

1. Crea un nuevo fork del proyecto del curso.  
URL del repositorio: <https://github.com/esancaro/Javascript-Course/>



2. Aboutme.js

```
JS index.js M X
Javascript-Course > activity > 1_session > JS index.js
1 //Ejercicio 1
2 import('./aboutme.js');
3 console.log("Ejercicio 1");
4
5 //Ejercicio 2

JS aboutme.js U X
Javascript-Course > activity > 1_session > JS aboutme.js > ...
1 const NOMBRE_APELLIDOS = 'Esteban Castro Rojas';
2 const PROFESION = 'Informático';
3 const PUESTO = 'Full-Stack Developer';
4
5 const MENSAJE = `Hola, me llamo ${NOMBRE_APELLIDOS} y me dedico a ${PROFESION}.
6 Estoy cursando este Máster porque me gustaría trabajar en ${PUESTO}`;
7
8 console.log(MENSAJE);

DevTools - 127.0.0.1:5500/activity/1_session/index.html
Identify your project's root folder to open source files in Visual Studio Code and sync changes. Set root folder Don't show again

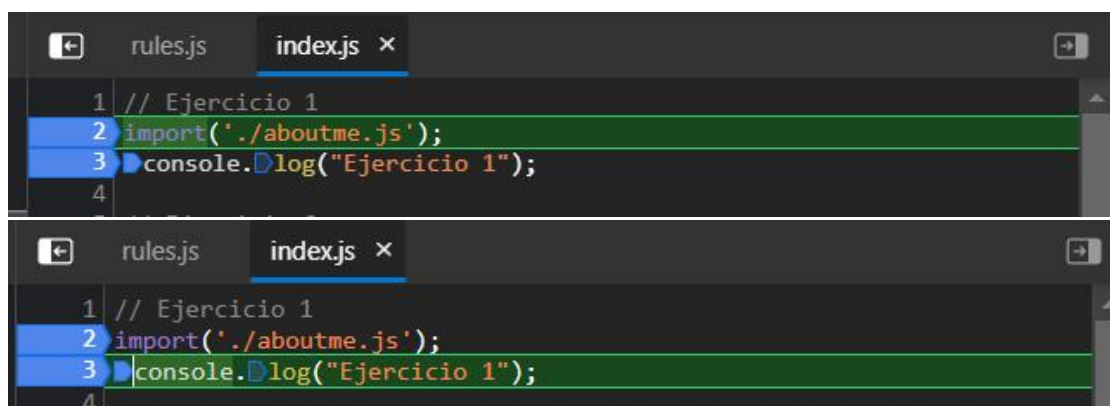
Elements Console Sources Network Performance Memory Application Security >> +
top Filter Default levels 2

Ejercicio 1
Hola, me llamo Esteban Castro Rojas y me dedico a Informático.
Estoy cursando este Máster porque me gustaría trabajar en Full-Stack Developer
>
```

## Ejercicio 2

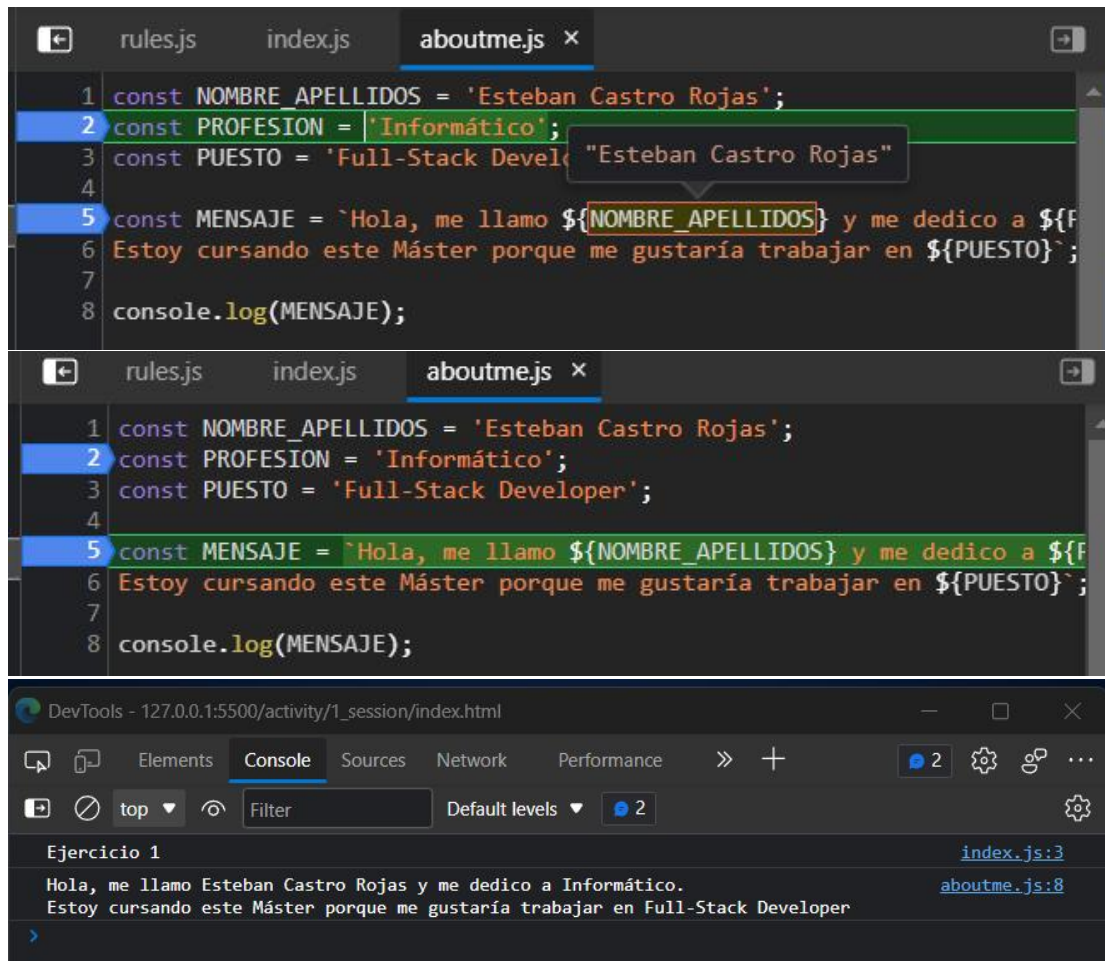
¿Qué se ejecuta antes, la llamada a `console.log()` o el contenido del fichero `aboutme.js`?

R/ Primero se ejecuta el `console.log`, después se ejecuta el código importado.



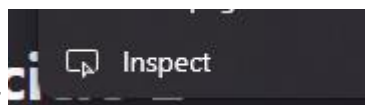
```
rules.js index.js X
1 // Ejercicio 1
2 import('./aboutme.js');
3 console.log("Ejercicio 1");
4

rules.js index.js X
1 // Ejercicio 1
2 import('./aboutme.js');
3 console.log("Ejercicio 1");
4
```



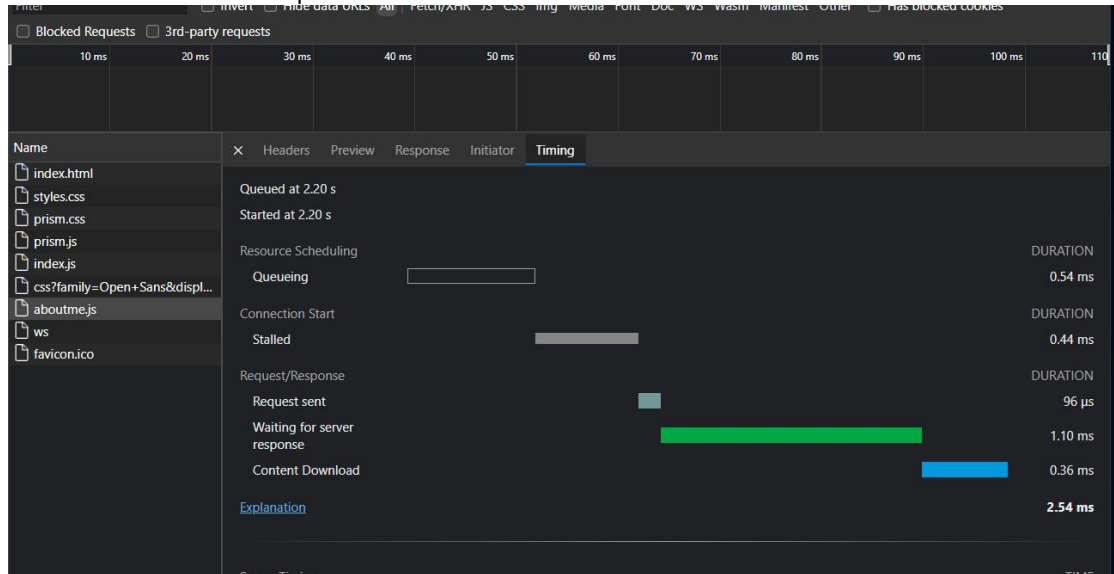
## Ejercicio 3

Enumera 3 herramientas útiles dentro de las herramientas de desarrollo de Google Chrome o Mozilla Firefox (inspector, consola, networking...) y explica su funcionamiento.



1. Inspector. Con el comando "inspect" se ejecuta la pantalla "Elements" que muestra el DOM del sitio web incluyendo el código de los iFrames. Tiene un apartado a la derecha donde se muestran los estilos CSS en contexto del elemento que se haya seleccionado así como "event listener" o escuchadores de eventos relacionados con el éste.
2. Consola. La consola provee una interfaz con el motor JS del navegador, permite ver las salidas de consola de los guiones que se estén ejecutando, así como los mensajes de error. También permite evaluar y correr código JavaScript sobre los objetos del DOM así como los módulos que hayan sido cargados.
3. Network. Esta tab muestra el detalle de cada página o recurso cargado por la sesión actual, ya sean páginas, elementos como imágenes o llamadas asincrónicas a recursos como servicios de información (REST). Cuando un sitio de internet se carga, el Network muestra el detalle de la solicitud, los encabezados, el contenido; el detalle de la respuesta con sus encabezados. Este último muestra la vista previa del cuerpo en la respuesta así como los datos en bruto. El diagrama de actividad de red muestra los tiempos de las solicitudes que están en espera, que están bloqueadas (estas están esperando un hilo de ejecución bloqueado por otras solicitudes). Muestra los tiempos para el envío de la solicitud, el tiempo que se toma el servidor en responder (en verde) y el tiempo en bajar el contenido, una vez resuelta la solicitud. Permite también

establecer el intervalo en que se desea ver resultados



## Ejercicio 4

Programa un script que imprima todos los números del 1 al 100 que sean divisibles por 7.

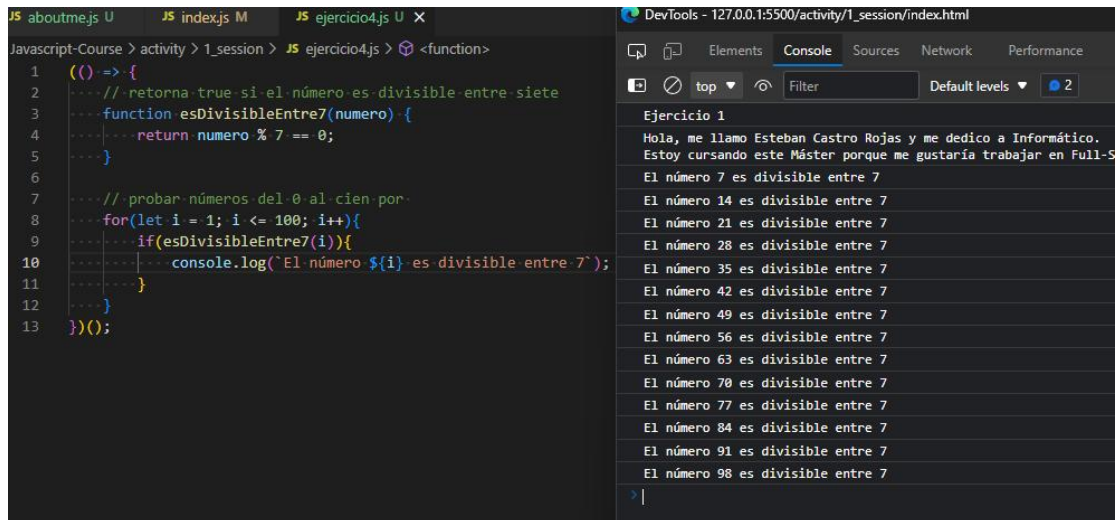
Código del ejercicio4.js localizado en el folder de 1\_session

```
(() => {  
  // retorna true si el número es divisible entre siete  
  function esDivisibleEntre7(numero) {  
    return numero % 7 == 0;  
  }  
  
  // probar números del 0 al cien por  
  for(let i = 1; i <= 100; i++){  
    if(esDivisibleEntre7(i)){  
      console.log(`El número ${i} es divisible entre 7`);  
    }  
  }  
})());
```

Import de ejercicio4.js desde index.js, línea 9.

```
// Ejercicio 4  
import('./ejercicio4.js');
```

Pantalla con el resultado de la corrida.



## Ejercicio 5

¿Cuál es la diferencia entre ejecutar un fichero JavaScript en un navegador web en formato `<script>` y en formato `<script type="module">`?

R/ La opción de tipo “module” le indica al navegador que el procesamiento del contenido debe ser aplazado<sup>1</sup> (defer). Esto significa que estos módulos son para uso de otros módulos, no para su inmediata ejecución. Los atributos “charset” y “defer” no son tomados en cuenta. Los módulos cargados por este medio requieren el uso del protocolo de origen cruzado CORS para obtenerlo (deben estar en el mismo dominio del cliente o este dominio debe estar declarado en el lado del servidor). Los guiones cargados con el atributo “nomodule” se utilizan como soporte en caso de que el navegador no soporte el atributo de tipo module.

## Ejercicio 6

Crea un objeto llamado “formatter” con un atributo, que llamaremos “prefix”, que tendrá de valor “Hello “, y un método que llamaremos “append”, que imprimirá la concatenación entre el atributo “prefix” y la cadena que pasemos como argumento en el método. Después, añadir mediante el atributo prototype otro método que acepte también un solo argumento e imprima esa misma cadena en minúsculas.

Código del ejercicio6.js localizado en el folder de 1\_session:

```
(() => {  
  const formatter = {  
    prefix: "Hello",  
    append: function(x) {  
      return `${this.prefix} ${x}`;  
    }  
  }  
  
  // 6: crear objeto, función, concatenar Hello World  
  console.log('Ejercicio 6:', formatter.append('World'));  
})
```

<sup>1</sup> Ref.: <https://developer.mozilla.org/en-US/docs/Web/HTML/Element/script>

```
// 6b: añadir método al proto
formatter.__proto__.toLowerCaseString = function(x){
    return this.append(x).toLowerCase();
}

console.log('Ejercicio 6b:', formatter.toLowerCaseString('Soy Esteban'));
})();
```

Salida:

```
El número 98 es divisible entre 7
Ejercicio 6: Hello World
Ejercicio 6b: hello soy esteban
>
```