CIP FP cecheste

Desarrollo Web en Entorno Cliente

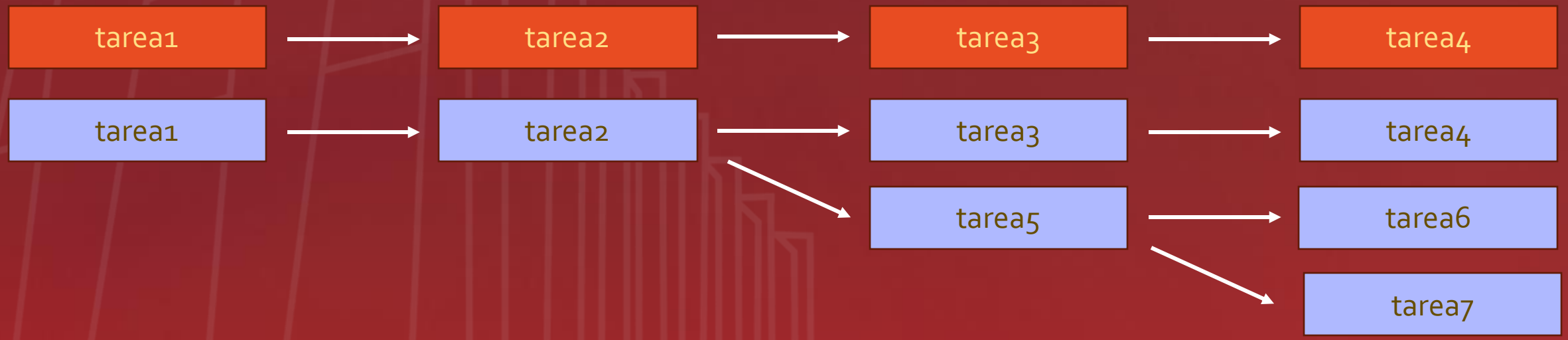
Javascript

Asincronía [1]

Asincronía

En programación, el concepto síncrono indica la ejecución secuencial de las instrucciones que incluimos en nuestra aplicación.

Por lo tanto, asíncrono será lo opuesto: acciones no bloqueantes que tienen que esperar a que ocurra algo que no depende de nosotros, que no sabemos cuando va a ocurrir o que ni si quiera va a ocurrir.



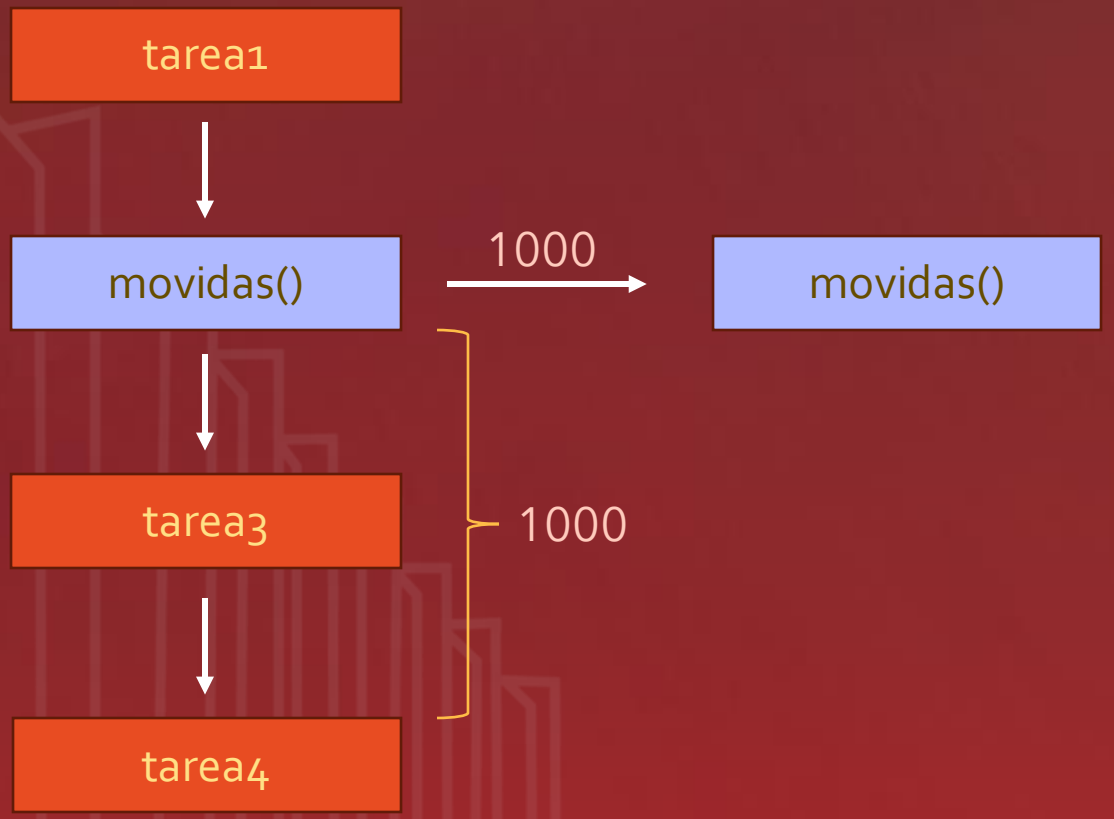
setTimeOut()

Función que ejecuta, de forma **asíncrona**, una función predefinida o no pasado un tiempo indicado en milisegundos.

```
function movidas(param1, param2) {
    console.log("movidas guapas " + param1 + param2);
}

setTimeout(movidas, 1000);

setTimeout(movidas);

setTimeout(movidas, 1000, param1, param2, ... paramN);

setTimeout((param1, param2) => {
    console.log("movidas guapas " + param1 + param2);
}, 1000, param1, param2, ... paramN);
```

setTimeout()

```
setTimeout(movidas, 1000);
```



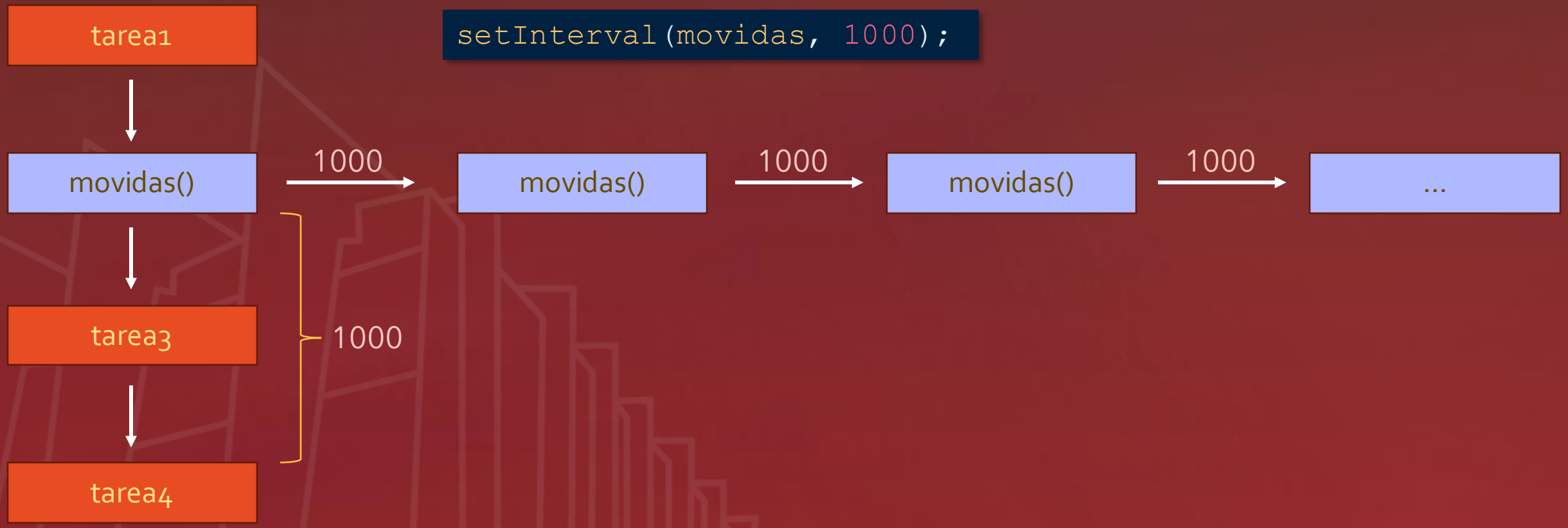
setInterval()

Función que ejecuta, de forma **asíncrona** y recurrente, una función predefinida o no pasado un tiempo indicado en milisegundos.

```
function movidas(param1, param2) {
    console.log("movidas guapas " + param1 + param2);
}

setInterval(movidas, 1000);

setInterval(movidas);

setInterval(movidas, 1000, param1, param2, ... paramN);

setInterval((param1, param2) => {
    console.log("movidas guapas " + param1 + param2);
}, 1000, param1, param2, ... paramN);
```

setInterval()

```
setInterval(movidas, 1000);
```



clearTimeout() y clearInterval()

En ambas funciones se devuelve un número entero que sirve para identificar el `setTimeout()` y el `setInterval()` correspondiente.

Si queremos cancelar cualquiera de los dos, deberemos emplear ese identificador para hacerlo. El sistema es el siguiente:

```
let idTO = setTimeout(movidas, 1000);

clearTimeout(idTO);

let idInt = setInterval(movidas, 1000);

clearInterval(idInt);
```

Prácticas

1. Realizar una aplicación que, pasado un tiempo aleatorio (de 2 a 5 segundos) muestre un div naranja con un botón que indique “Púlsame” en un lugar aleatorio de la pantalla. El usuario deberá pulsar el botón cuando lo vea y el programa, mediante un `alert()`, mostrará el tiempo que le ha costado pulsarlo. Al aceptar, volverá a suceder.

2. Realizar un programa al que le pasemos un número de minutos para el fin del mundo mediante un formulario, y cree un cronómetro con el formato HH:mm:ss:ds:cs, que finalice cuando llegue a cero.