**setTimeout(function, miliseconds)**

* Ruleaza odata, dupa ce timpul dat a expirat, adica e programat sa ruleze peste un inverval de timp
* Functia data creaza un nou thread, deci main thread nu va astepta ca acesta creat sa se execute, ci va executa codul mai departe
* setTimeout(() => *console*.log('Executed after 5 seconds'), 5000);  
    
  *console*.log('Nu va astepta ca functia de sus sa ruleze')

Nu va astepta ca functia de sus sa ruleze

Executed after 5 seconds

* Exista totusi o problema, si anume ca nu putem sa trimitem parametrii functiei, dar problema poate fi rezolvata asa:

**setTimeout(function, miliseconds, param1, param2 ,...)**

Deci, argumentele dupa miliseconds vor fi trimisi functiei

setTimeout((param1, param2) => *console*.log(param1, param2), 5000,'Hi','My name is');

* Odata ce timerul e pornit, putem sa anulam executia lui cu **clearTimeout(ref)**
* const *timeout* = setTimeout((param1, param2) => *console*.log(param1, param2), 5000,'Hi','My name is');  
  clearTimeout(*timeout*);

**setInterval(function, miliseconds)**

* Ruleaza o functie de mai multe ori, odata la un inverval stabilit
* setInterval(() => *console*.log('Hello'), 2000)

mesajul va fi executat la infinit in fiecare 2 secunde

* **clearInterval(obj)** – opreste obiectul interval