Timers in Javascript

How to delay the execution of a function

Delaying the execution of a function

- Timer functions can be used to delay or repeat the execution of other functions (which they receive as their first argument).

- There are two methods for it:
  - ***setTimeout*** allows us to run a function once after the interval of time.
  - ***setInterval*** allows us to run a function repeatedly, starting after the interval of time, then repeating continuously at that interval.

- These methods are not a part of JavaScript specification. But most environments have the internal scheduler and provide these methods. In particular, they are supported in all browsers and Node.js.

IT TALENTS
Training Camp

setTimeout()

- Syntax:

```
setTimeout(function, milliseconds, param1, param2, ...)
```

- Parameter Values

Parameter	Description
Function	Required. The function that will be executed
milliseconds	Optional. The number of milliseconds to wait before executing the code. If omitted, the value 0 is used
param1, param2, ...	Optional. Additional parameters to pass to the *function* (Not supported in IE9 and earlier)

IT TALENTS Training Camp

setInterval()

- Syntax:

```
setInterval(function, milliseconds, param1, param2, ...)
```

- Parameter Values

Parameter	Description
function	Required. The function that will be executed
milliseconds	Required. The intervals (in milliseconds) on how often to execute the code. If the value is less than 10, the value 10 is used
param1, param2, ...	Optional. Additional parameters to pass to the *function* (Not supported in IE9 and earlier)

Demo

IT TALENTS
Training Camp

Cancelling Timers

- Because calling a timer function schedules an action, that action can also be cancelled before it gets executed.

- A call to setTimeout returns a timer “ID” and you can use that timer ID with a clearTimeout call to cancel that timer. Here’s an example:

```
const timerId = setTimeout(
  () => console.log('You will not see this one!'),
  1000
);
clearTimeout(timerId);
```

- Cancelling setInterval can be done using the ***clearInterval(id)*** function

Time delay is NOT guaranteed!

- Try to execute to following code:

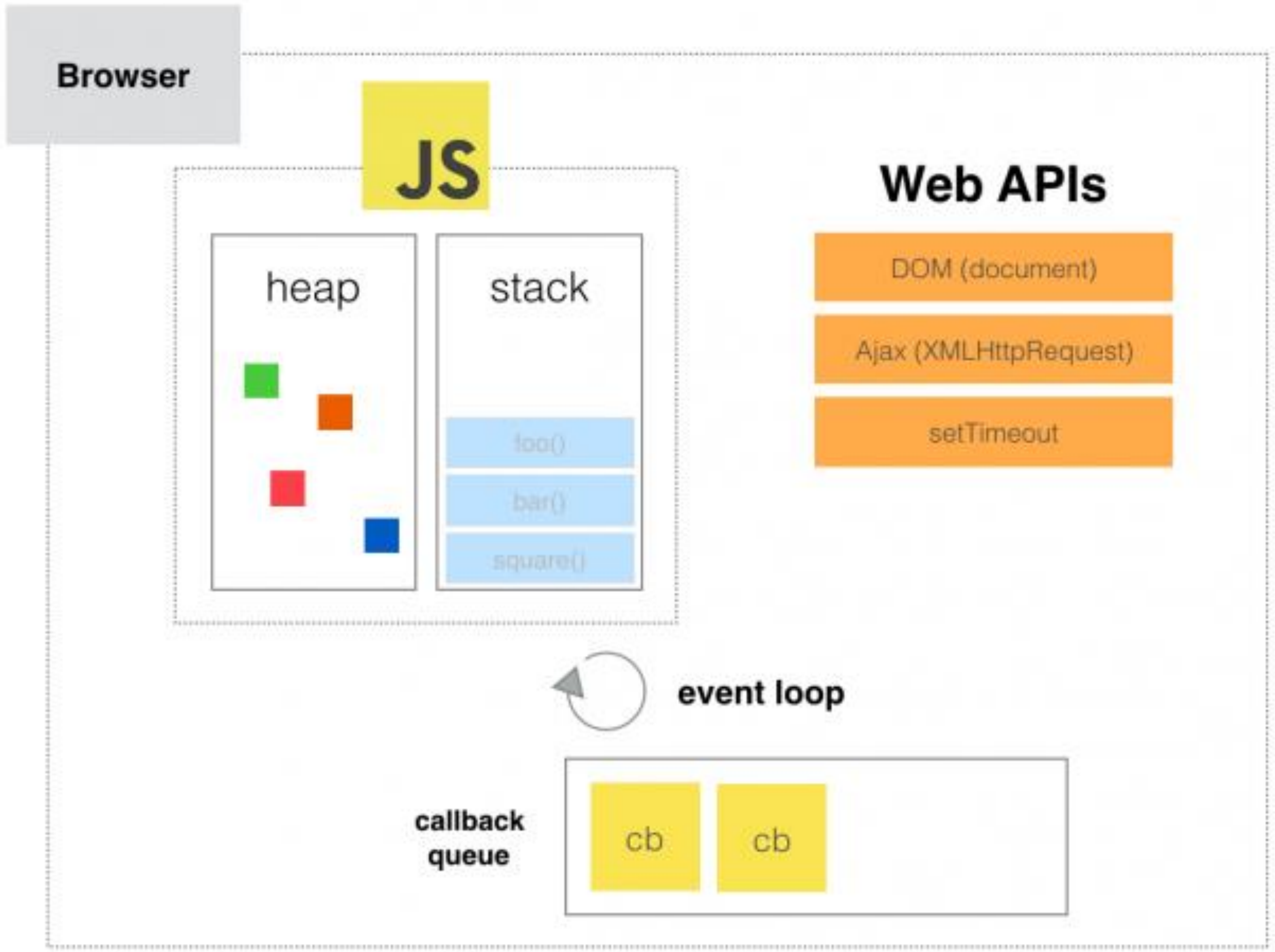
```
console.log(1);

setTimeout(() => console.log(2), 0);

console.log(3);
```

- What is happening?

Event Loop


Browser
JS
heap
stack
foo()
bar()
square()
Web APIs
DOM (document)
Ajax (XMLHttpRequest)
setTimeout
event loop
callback queue
cb
cb

Tricky questions

```
// What will the following code output?
const arr = [10, 12, 15, 21];
for (var i = 0; i < arr.length; i++) {
  setTimeout(function() {
    console.log('Index: ' + i + ', element: ' + arr[i]);
  }, 3000);
}
```

```
// In what order the numbers will be printed?
 console.log(1);
 setTimeout(function(){console.log(2)}, 1000);
 setTimeout(function(){console.log(3)}, 0);
 console.log(4);
```

```
// What will the following code output?
for (var i = 0; i < 5; i++) {
    setTimeout(function() { console.log(i); }, i * 1000 );
}
```

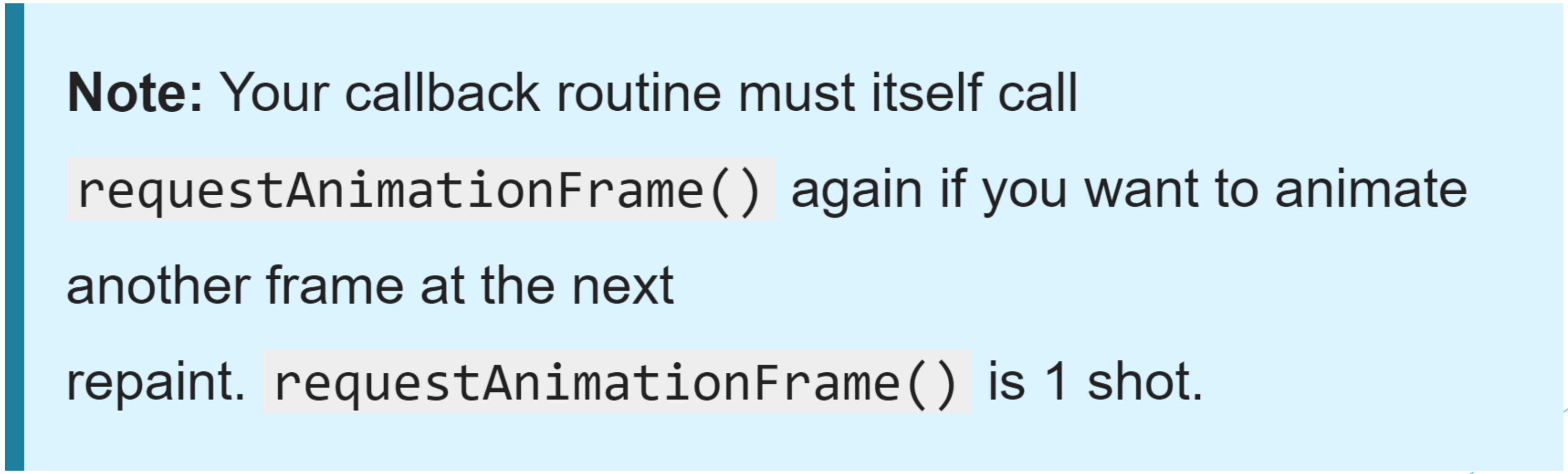
IT TALENTS
Training Camp

If the time is not guaranteed, how all JS animations are made?

IT TALENTS
Training Camp

requestAnimationFrame()

- The window.requestAnimationFrame() method tells the browser that you wish to perform an animation and requests that the browser calls a specified function to update an animation before the next repaint. The method takes a callback as an argument to be invoked before the repaint.



Note: Your callback routine must itself call `requestAnimationFrame()` again if you want to animate another frame at the next repaint. `requestAnimationFrame()` is 1 shot.

Which one should I choose?

setTimeout()

- The execution time is NOT guaranteed.
- The execution can be easily stopped.

requestAnimationFrame()

- The scheduling is much more accurate than setInterval() and setTimeout().
- Multiple animations are possible using requestAnimationFrame.
- CPU optimization is done using requestAnimationFrame by not drawing when tab or window is not visible.

IT TALENTS
Training Camp

Resources

- You will find hundreds of videos on these topics, but the most useful and mandatory for you are:
  - Event Loop: https://www.youtube.com/watch?v=8aGhZQkoFbQ&feature=emb_logo
  - SetInterval vs RequestAnimationFrame – video: https://www.youtube.com/watch?v=8aGhZQkoFbQ&feature=emb_logo
  - setTimeout vs RequestAnimationFrame – article: https://medium.com/javascript-in-plain-english/better-understanding-of-timers-in-javascript-settimeout-vs-requestanimationframe-bf7f99b9ff9b

IT TALENTS
Training Camp