Web Workers

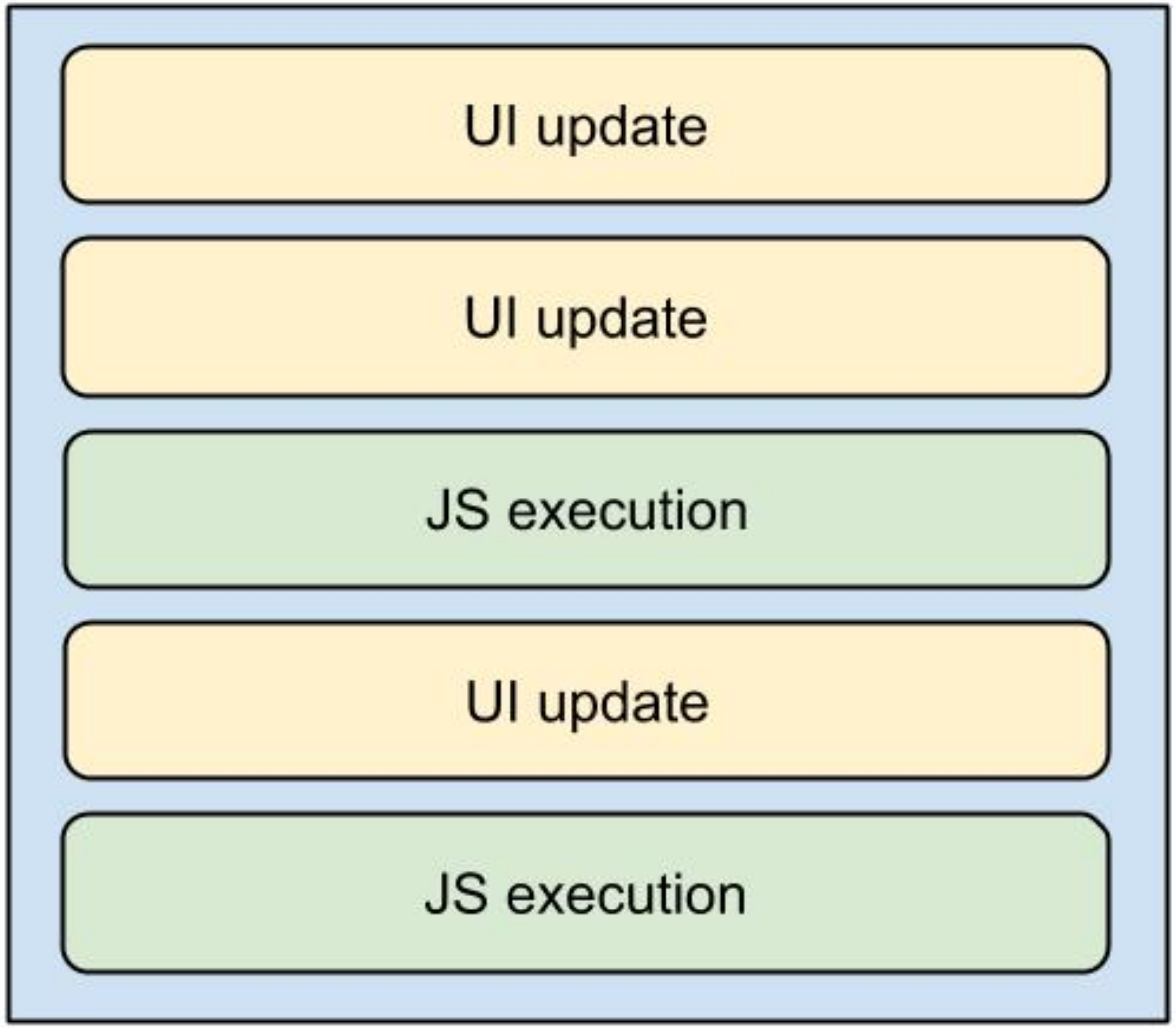
Web workers are ~~a way of pretending we have~~ **threads you can use in browsers.**

Web workers **are actually OS level threads.**

Web workers **provide a way to run JavaScript work outside of the main browser event loop (UI thread).**

Huh?

browser event loop



UI update

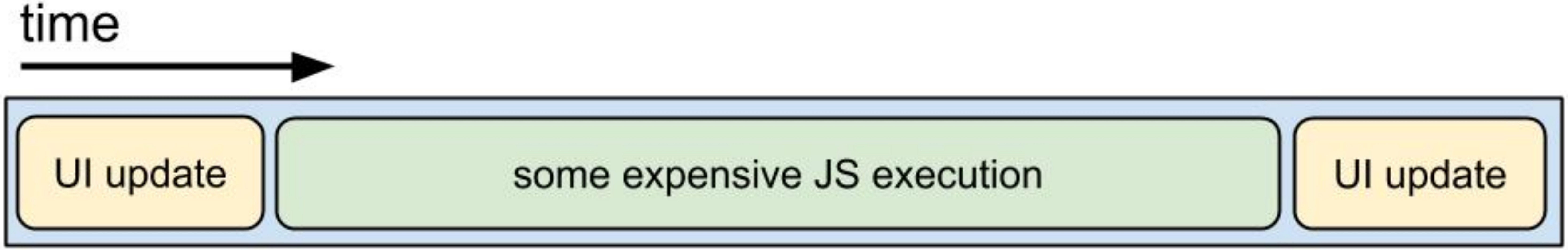
UI update

JS execution

UI update

JS execution

example UI thread timeline



time

UI update

some expensive JS execution

UI update

execution bottlenecks = slow UI

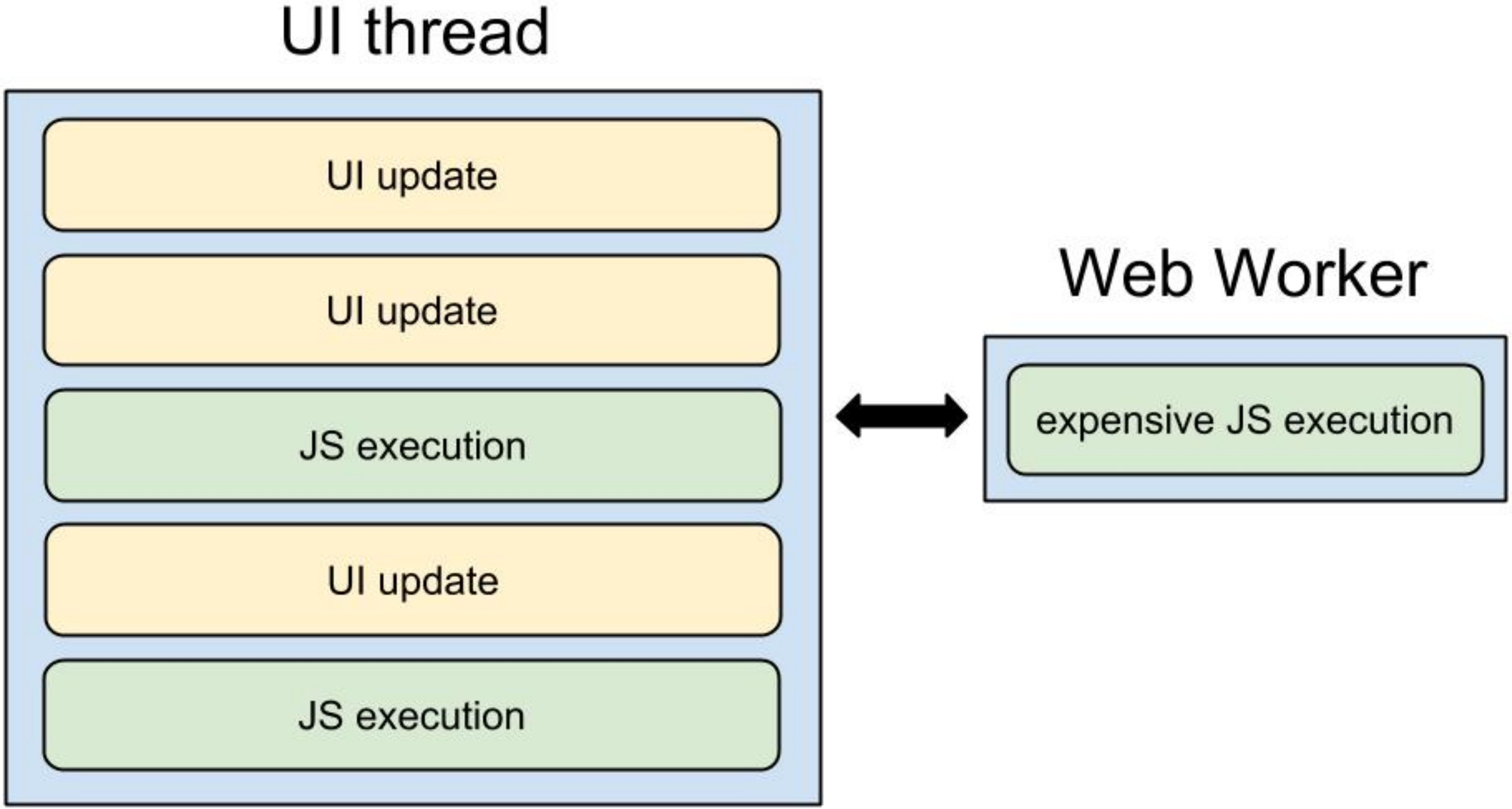
🐢

demo

slow UI = sad UX

😢

what can we do?



UI thread

UI update

UI update

JS execution

UI update

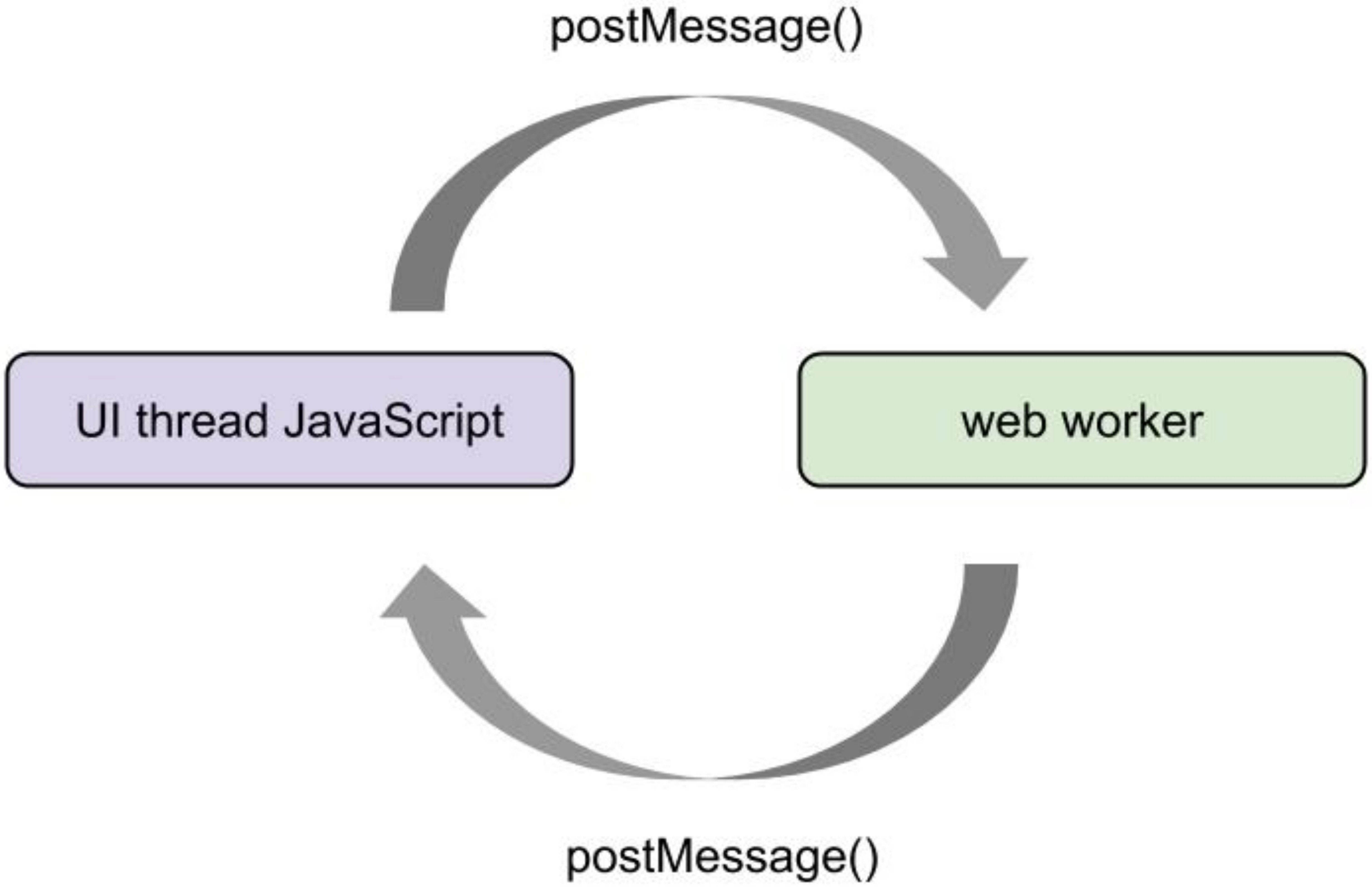
JS execution

Web Worker

expensive JS execution

😍 🐇

how does it work?



postMessage()

UI thread JavaScript

web worker

postMessage()

Example

```
// from the UI thread

// create new worker, point to worker file
var myWorker = new Worker("worker.js");
```

Example

```
// from the UI thread

// create new worker, point to worker file
var myWorker = new Worker("worker.js");

// send strings, arrays, numbers, objects...
myWorker.postMessage([1, 2]);
```

Worker Example

```
// from worker.js

onmessage = function(e) {
  var workerResult = 'the sum is ' + (e.data[0] + e.data[1]);
  postMessage(workerResult);
}
```

Back to the UI thread

```
// message listener
myWorker.onmessage = function(e) {
  console.log(e.data); // 'the sum is 3'
}
```

Back to the UI thread

```
// message listener
myWorker.onmessage = function(e) {
  $('.myDiv').html(e.data);
}
```

Back to the UI thread

```
// create new worker, point to worker file
var myWorker = new Worker("worker.js");

// send strings, arrays, numbers, objects...
myWorker.postMessage([1, 2]);

// message listener
myWorker.onmessage = function(e) {
  $('.myDiv').html(e.data);
}
```

demo

Can I use libraries within Web Workers?

Yes!

```
// within worker.js
importScripts('underscore.js', 'handlebars.js');
```

How do I get rid of a Worker when I no longer need it?

How do I get rid of a Worker when I no longer need it?

```
// from UI thread
myWorker.terminate();
```

Slightly better

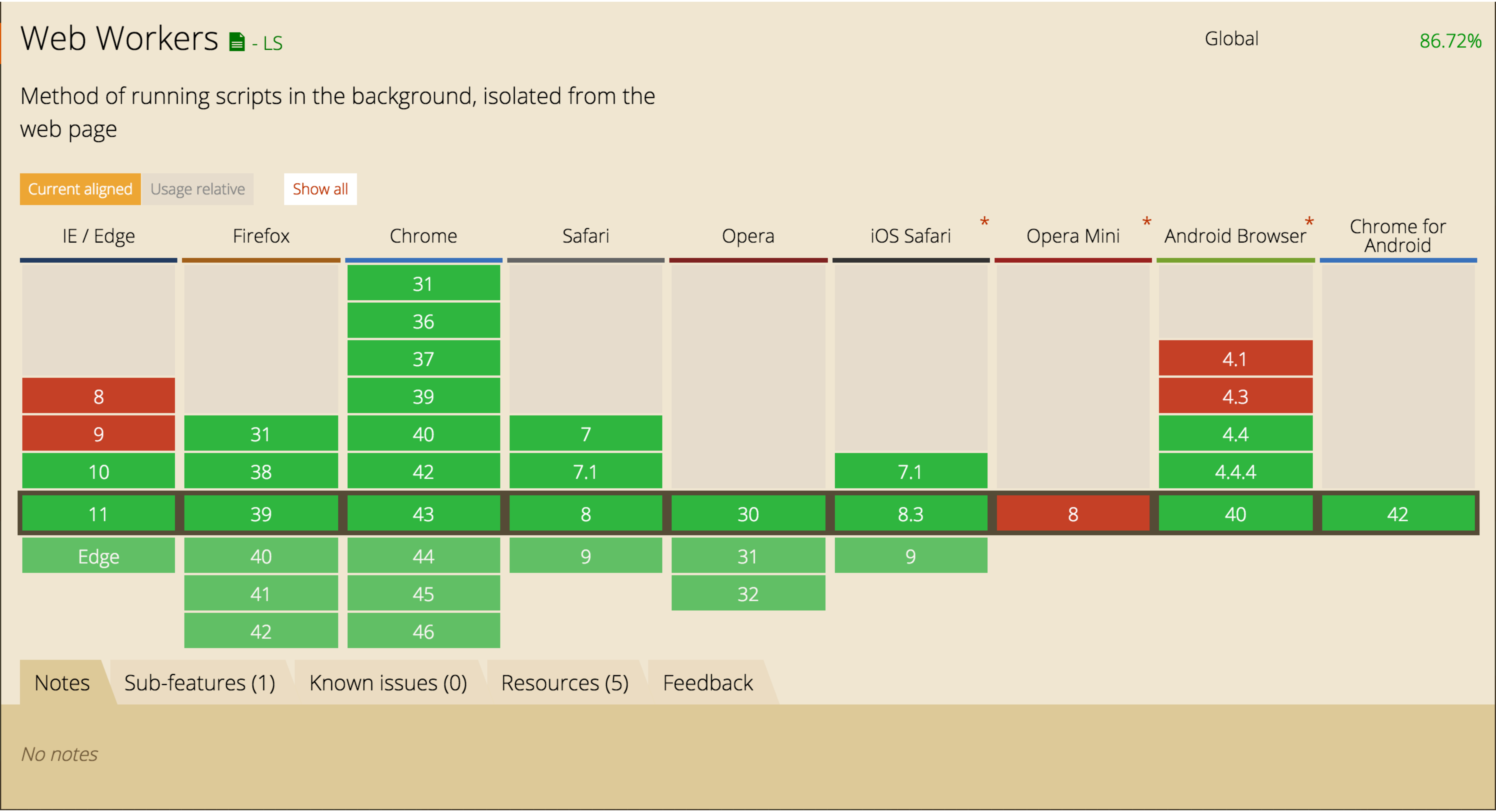
```
// from UI thread
myWorker.postMessage('worker:close');


// from worker.js
onmessage = function(e) {
  if (e.data === 'worker:close') {
    close();
  }
}
```

Can I spawn a Web Worker from a Web Worker?

YES YOU CAN

Limitations



Web Workers - LS

Global 86.72%

Method of running scripts in the background, isolated from the web page

IE / Edge	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
		31						
		36						
		37					4.1	
8		39					4.3	
9	31	40	7				4.4	
10	38	42	7.1		7.1		4.4.4	
11	39	43	8	30	8.3	8	40	42
Edge	40	44	9	31	9			
	41	45		32				
	42	46						

No notes

No cross origin shenanigans

No DOM access

(AJAX is cool though)

You **can** crash the browser if you spawn too many Workers.

Multithreaded Programs in JavaScript

Web Workers

O'REILLY®

Ido Green