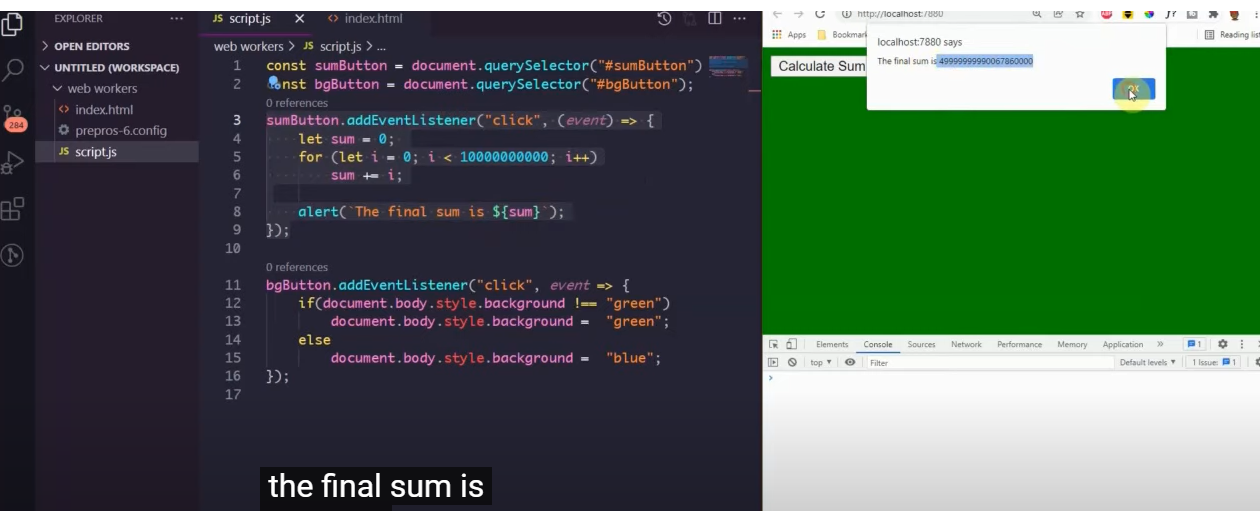
[JavaScript Web Workers Explained](https://www.youtube.com/watch?v=Gcp7triXFjg)

Web worker is simply just a JavaScript process that runs in the background.

JavaScript, by default is a single threaded language, the main thread is responsible for all the processing that happens, which executes the code linewise.

A web worker is another JavaScript thread, which allows multiple JS threads to run in parallel with each other. Instead of main thread doing all the heavy lifting work, the web-workers can be created to perform any computationally expensive work, so that the main thread doesn't get blocked & can perform other important stuff.

|  |  |
| --- | --- |
| **Main Thread** | **Web worker** |
| * 1. Only the main thread can manipulate & access the DOM. | * 1. No DOM Manipulation |
|  |  |



Here, we can't execute back ground change listener after clicking on perform sum action.

Reason - The sum () takes longer to execute & hence the more wait!! Since, all of this is now running on the main thread, hence, the other executions are blocked.

So, we can create a web-worker for the heavy task - eg - perform sum.

Steps -

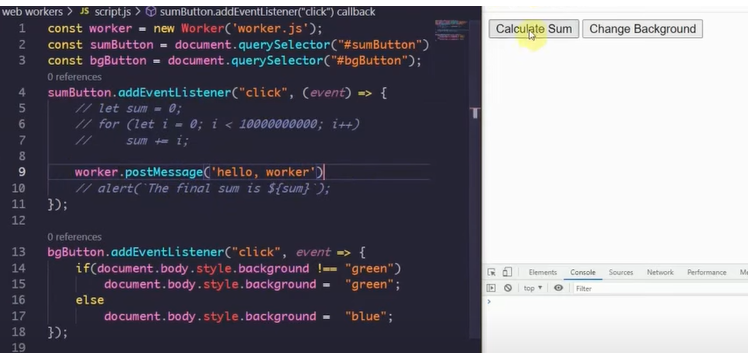
1. Create instance of the worker.

const worker = new Worker('worker.js');

1. Create a new worker.js script.
2. How to send data from main script (script.js) to worker.js?

=> 'Post Message' api does that for us.

=> It raises an event from one script to another script to catch or listen to.



Text

Description automatically generated

Text

Description automatically generated

Now, the worker is doing all the heavy lifting and when a user tries to toggle b/w the background, he'll be able to do that.

1. To pass a message from the worker to the main thread - create an instance of the worker method being called in the main thread.
2. 