Using dedicated workers for background work

altimin@chromium.org
Jun 19, 2018
PUBLIC

DRAFT

Overview

Now dedicated workers associated with backgrounded pages are unrestricted and this results in dedicating workers accounting for 20% of renderer process CPU, which rises to 43% for background tabs.

We want to throttle and freeze dedicated workers in the background tabs which, as early experiments show, have a significant positive impact on foreground tabs (page load and scroll latency metrics improvements). However there is a variety of legitimate use cases for dedicated workers in background tabs (including SETI calculations, bitcoin mining, hosting multiplayer games, processing orders for shop PWA).

Proposal

We propose to use dedicated workers as a long-term solution for doing complex processing for background web apps and require an explicit user permission to do this.

Dedicated workers allow a web app to implement arbitrary processing and maintain network connections. Chrome benefits from the ability to force background work off the main thread and more aggressively throttle / freeze / discard pages in the background, saving CPU and memory without significant breakage.

Implementation & launch

A new parameter will be added to Worker's constructor:
```
worker = new Worker('script.js', { 'background_execution': true })
```

Page may query the result by the newly-added `worker.background_execution_allowed` field, which will be set to true iff the Origin Trial is enabled and the 'background_execution' flag is passed to the Worker's constructor.

'background_execution' flag requests background execution privileges for this worker. Dedicated workers without this privilege will be throttled and frozen while in background. The default value of 'background_execution' flag is false, therefore it serves as an opt-out from enabled-by-default intervention.

It is proposed to launch this behind Origin Trials, requiring web developers to enroll into the Origin Trial and specify the motivation for this ‘background_execution’ flag is, effectively, an opt-out from the dedicated worker throttling and freezing interventions.

Note that this flag is expected to remain and become a part of Web Platform, as opposed to opt-outs from main thread interventions, which are expected to be gradually removed.

Further plans

The following data will be monitored as a part of the launch:

- Number (and list) of origins using this feature
- List of use cases provided by the origin during OT enrollment process
- The amount of work coming from workers with background execution permission and its impact on the foreground tab experience.

The longer-term plans include UX change to notify the user when page does this background processing (“badging” similar to Android foreground service notifications). The user permission is required to distinguish between legitimate user-sanctioned heavy computations (including cryptocurrency mining) and heavy computations performed by malicious apps / ads (including cryptocurrency mining).

The exact details will be discussed with UX review and implemented before making the option default (non guarded by Origin Trials).

Depending on the UX studies and the data from the field a proactive permission might be necessary, which is a subject for further discussion.

Frequently asked questions

Why this proposal concerns only workers and does not touch the main thread?

Doing work on the main thread in the background has a lot of disadvantages — a wide API surface can be used by the page, including DOM access and the difficulty to distinguish between accidental work that should be done in the foreground only and does not check for visibility status and the truly background work. These disadvantages make memory (keeping DOM active) and CPU (non-required work, expensive layouts, expensive GC) prohibitive and with Lifecycle web moves to stopping all main thread work as soon as possible.

Why is this different from the Lifecycle interventions opt-outs?

Lifecycle interventions target main thread work and are designed to be temporary and to be gradually removed after some time.

This proposal involves adding a new API surface, which is planned to become a permanent part of the Web Platform.