Blink Scheduler friendly HTMLDocumentParser

kouhei@

[PUBLIC DOCUMENT]

tl;dr:

HTMLDocumentParser should load page fast while not causing janks, or undesired frame drops, by hogging Blink main thread. Blink Scheduler improves scheduling of tasks in main thread by prioritizing certain tasks. HTMLDocumentParser should use Blink Scheduler explicitly and schedule its tasks in right priority so that it would not cause janks while keeping page load fast.

Current Status as of Mar. 24, 2015:

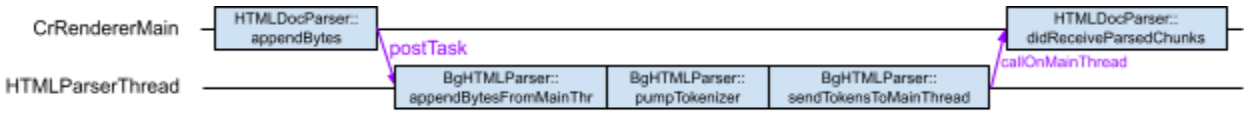
kouhei@ landed key CLs for high priority tasks yielding and fine grained task. alexclarke@ made parser tasks posted as loading tasks.

Background

How does HTMLDocumentParser currently work?

HTMLDocumentParser is a component in Blink used to process HTML. It receives HTML byte stream and outputs a DOM tree.

HTMLDocumentParser parses HTML documents asynchronously when possible[1]. HTMLDocumentParser delegates decoding/tokenizing to background parser run on a separate thread, and then consumes the tokens progressively to generate DOM tree.



When HTMLDocumentParser receives a new data chunk from loader, it delegates decoding/tokenizing by posting the new data chunk to `HTMLParserThread`. The `BackgroundHTMLParser` run on the parser thread processes it and returns the result token chunk to main thread.

The decoded token chunks are then progressively consumed in main thread. `HTMLDocumentParser` stores received token chunks in a queue, and call `pumpSpeculations` to processes the chunks in the queue. `pumpSpeculations` processes the queue until a time limit of 500ms is reached or it encounters a script which must run synchronously. If time limit is reached while pumping, it schedules continuation by scheduling a one-shot timer task, and yields main thread.

Task ordering constraints

Any task may be scheduled to be run before / after a pumpSpeculations task. There is no task ordering constraint.

- Any task may put to run before a pumpSpeculations task.
- A pumpSpeculations task may run blocking <script> tag.
    - Tasks scheduled as a result of javascript execution are currently executed before the next pumpSpeculations task.
    - ^^^ However this behavior isn't in the spec. The spec allows using arbitrary number of task queues. Ordering constraints only apply to tasks from the same task source.

Blink Scheduler

Blink Scheduler is a new task scheduler for Blink main thread. Before Blink Scheduler was introduced to Blink, all tasks were executed in order which they were queued. Blink Scheduler enables more sophisticated task scheduling with priorities and deadlines. Input and compositor

[1] HTMLDocumentParser is forced to process HTML synchronously when triggered from `document.write`, `innerHTML`, etc.

tasks are prioritized over other tasks, and allows low-priority tasks to be skipped while user is interacting with the web page.

Goals

Display above-the-fold region of new web pages as fast as possible.

The above-the-fold region is the region visible to user without scrolling the page (Diagram 1). It is known that users do not scroll web pages 75% of the time. We should display this region as fast as possible for better user experience.

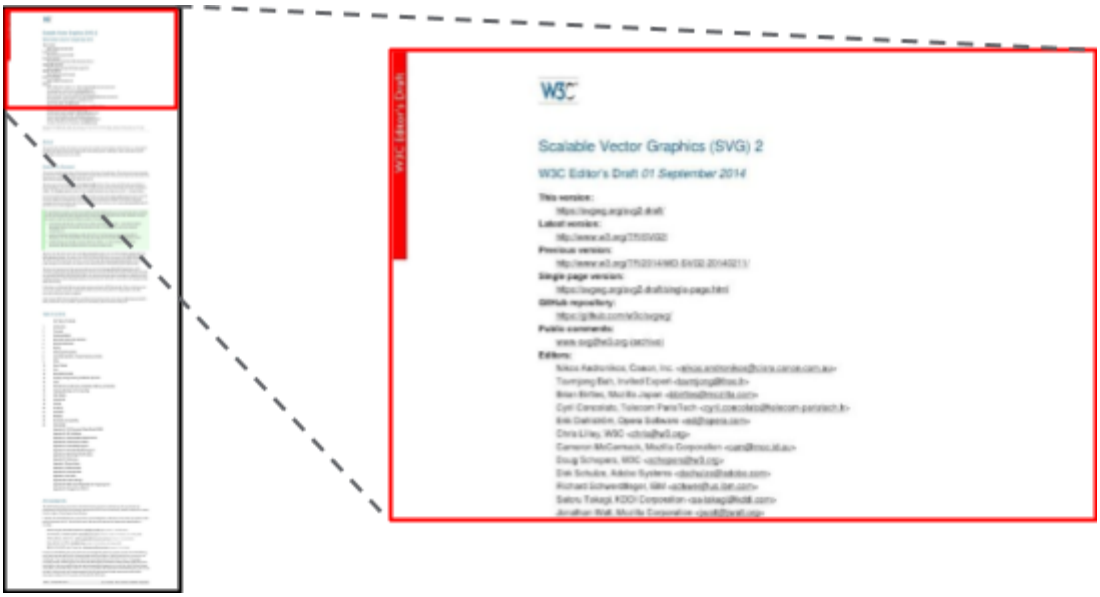


Diagram 1: Above the fold region

Short page load time (PLT) == Blink should load web pages fast.

Page load time (PLT) is defined as time taken load all resources of a web page. We should keep this fast for user experience and for various benchmarks. PLT is often the time user is required to wait before any interaction with the web page. PLT is a common metric to measure browser performance. For example, Tom's hardware reviews use this metric to compare browser performance.

HTMLDocumentParser should not cause janks after page is loaded.

HTMLDocumentParser should not cause janks while processing additional HTML contents loaded after initial page load. HTMLDocumentParser is also invoked after the initial page load for loading additional content via XMLHTTPRequest. These loads typically occur concurrently with "loading spinner" animation and additional user inputs are allowed while loading. Animation and handling user input should be responsive, and HTML parsing should not block these tasks.

TODOs / Implementation Ideas

Identify the key metrics.

We should inherit metrics proposed in Blink Scheduler doc.
TODO: any additional metrics to cover?

Make HTMLDocumentParser yield more often.

Landed: https://codereview.chromium.org/673603002/
Blocked on: key metrics
Currently, `HTMLDocumentParser` tries to `pumpSpeculations` inline after receiving new data chunk from `BackgroundHTMLParser`. This means that we `pumpSpeculations` in IPC task priority. We should remove the inline invocation of `pumpSpeculations` and have Blink Scheduler schedule a separate task for pumping it.

HTMLDocumentParser should schedule "process token chunks" task using Blink Scheduler

Currently, HTMLParserScheduler uses a Blink::Timer to schedule a task to process token chunks received from BackgroundHTMLParser. This should be changed to use Blink Scheduler task explicitly.

HTMLDocumentParser tasks should have high priority during initial page load, and low priority after initial page load.

We might want to use different priorities for scheduling parser tasks depending on page loading state: Before first paint, after first paint, after load complete.

- When we start loading page, we want to render its first frame as fast as possible by prioritizing compositor tasks, as it would likely cover the above the fold region.
- After completing the first paint, we want complete the load as fast as possible, thus we want to prioritize running the parser over compositor tasks, as compositor tasks will cause excessive recalcStyle/layout/paint which will be immediately thrown away by additional contents added by parser.
- After load is complete, we can prioritize input/compositor tasks again. Additional contents load will trigger HTMLDocumentParser, but loading is typically triggered concurrently with handing user input and/or "loading spinner" animation which should be prioritized over parsing for better user experience.

Possible Signals

- Above the fold rendering complete
- FirstPaint of the FrameView
    - Can be used to approximate "above the fold rendering complete"

- DOMContentLoaded for main frame Document
    - Indicates parser is done processing main Document. We can lower the priority of parser tasks after this.
- Input event in last X ms.
    - Indicates user is interacting with the page. Chrome should prioritize generating new compositor frames over loading additional contents.

CLs

- r183407: Yield HTMLDocumentParser for high priority tasks in blink scheduler.
-