Scheduler Overhaul, Phase 2

skyostil, brianderson, sievers, nduca, klobag

Status: Deprecated. Ideas are valid but priorities and shape have changed.

This doc assumes that we have done phase 1 of the scheduler overhaul. This is a dumping ground, we haven't gotten this to a design doc quality level yet. :)

Goals

On **all** platforms, with **one** architecture, we wish to:

- Maximize throughput of pure rAF-based applications, e.g. JSGameBench, MapsGL or Pong. See tradeoff note below.

Phase 1 gets us pretty far. But, rAF-driven benchmarks may not be able to get peak throughput given the Phase 1 architecture.

Also, by delivering input events only on VSync, we no longer are able to send high frequency input events on high resolution mice and tablets. To make that work, we want to modify the RenderWidgetHostInputController and its clients to do less input coalescing. Instead, the coalescing should be done at the injection point into WebViewImpl and/or CCLayerTreeHostImpl.

Max-Throughput rAF

https://docs.google.com/a/google.com/drawings/d/1XYZaWukA5Qygywgxjea_JSi_tvi7lAbvCtm2p6DT-qg/edit?usp=sharing

Draw immediately after Pending Tree Activation

In Phase 1, drawing on the impl thread only occurs between a BeginFrame and the Draw deadline. However, if there is no user input consumed on the Impl thread, a draw immediately after a tree activation (that came after a Draw deadline) will be no different than the draw at the next BeginFrame.

The con:

1. The one negative of drawing early is that it may push back the processing of the next BeginFrame, and thus RAF.

The pros:

1. By drawing immediately we might still make the Browser's deadline, if we are lucky.
2. Even if we do miss the Browser's deadline, performing the draw early will leave time later for other tasks.
3. If we don't tie RAF to BeginFrame, as proposed later in this document, drawing is less likely to push back RAF.

Deserializing RAF + Rasterization

RAF and rasterization of the pending tree are currently serialized because RAF is only triggered if the pending tree has been activated, which can only happen if the pending tree has been rasterized. For scheduling purposes, we should think of Rasterization as a stage between the Main thread and Compositor thread: we should be able to get pipelined parallelization between RAF and rasterization. To enable this concept, we have two options:

1. Add a commit tree that sits before the pending tree. Adding the commit tree will allow a second commit to complete before the previous commit's rasterization has completed.
2. Send RAF early (even if we have a pending tree) and block the second commit from completing until the pending tree has been activated.

The second option is conceptually simpler, but blocks the Main thread more often than the first option.

Optimistic RAF: Does RAF need to come at regular intervals?

Some dev's might assume RAF comes at regular intervals. Would it be bad to break that assumption? If not, the Impl thread can send RAF to the Main thread before it receives BeginFrame from the Browser thread - for example, on pending tree activation if the main thread is not consuming any input.

Throttling Optimistic RAF: Early feedback that the renderer missed the deadline

In Phase 1, the Browser throttles the Renderer simply by skipping a BeginFrame. We need another mechanism for the Renderer to better know how to throttle Optimistic RAF. If the Browser sent a message to the Renderer when the Browser draws regarding whether a new Renderer frame was used or not, that could be used to skip an Optimistic RAF and avoid adding an additional frame of latency because we produced an extra frame.

```
class OutputSurfaceClient {
 void beginFrame(TimeTicks draw_by_deadline);
 void frameMissedDeadline();
}
```

RAF Deadline

A RAF deadline is how long after a BeginFrame the Impl thread will still send a RAF to the Main thread, if tree activation occurs after a BeginFrame. The RAF deadline is not the same as the Draw deadline. The RAF deadline should depend on the expected Main thread response time, whereas the Draw deadline depends on the expected Impl thread response time.

```
Scheduler:
  setRAFDeadline(TimeTicks deadline
```