[wip] Viewporting in Blink

ajuma@, awoloszyn@, schenney@, vollick@

Background

The viewport is currently known only to cc, not to Blink. Blink notifies cc about invalid rects, and cc intersects these with an expanded version of the viewport in order to determine regions that need to be re-recorded by Blink. With knowledge of the expanded viewport, Blink will be able reduce the work it does when painting, since RenderObjects entirely outside this region will be able to be skipped. Further, Blink will be able to "push" recordings to cc, rather than having recordings "pulled" by cc from Blink. This will simplify logic in Blink and in cc. Moving invalidation tracking to Blink will allow invalidations to be tracked in a more general way; rather than necessarily tracking spatial regions, Blink will be able to track individual properties that are dirty, potentially allowing for faster re-recording where only stale draw-ops get updated.

Assumptions

We assume that cc is using impl-side painting.

The current state

The viewport

cc combines its knowledge of viewport size, page scale, scroll positions, and transforms to determine what's currently visible in the viewport. This happens in CalculateDrawProperties, which computes a visible rect in content space for each layer.

Invalidation

When an invalidation occurs in Blink, a dirty rect is sent to cc. These rects are sent by RenderLayerRepainter::setBackingNeedsDisplayInRect, and received by cc::Layer::SetNeedsDisplayRect. A set of dirty rects for each layer is maintained by cc. Before recording, cc simplifies the set of dirty rects int a smaller set of larger rects.

Note that invalidations caused by composited scrolling are handled the same way as invalidations caused by web content. cc doesn't track such invalidations separately. Instead, updated scroll offsets are sent to Blink (e.g. to ScrollableArea via WebLayerScrollClient). This triggers layout and invalidation in the same manner as Blink-driven changes to scroll position.

Recording

Recording happens during cc::LayerTreeHost::UpdateLayers. More specifically, recording for a single layer happens in cc::PicturePile::Update.

When deciding what to record for a layer, the layer's visible rect (that is, the bounds of the layer intersected with the viewport bounds, expressed in the layer's own space) is expanded by 8000 pixels in every direction. The dirty rects are intersected with this expanded visible rect to determine the rects that need to be recorded. However, frequently invalidated tiles (tiles that have been invalidated 75% or more of the time over the previous 32 frames) that **aren't** close to the (non-expanded) visible rect (at least 512 pixels away from the visible rect) aren't recorded.

Android WebView is handled differently, due to constraints imposed by the WebView API -- the entire layer is always recorded.

Other uses of the viewport in cc

The viewport is (at least implicitly) used on the compositor thread for touch hit testing and for rasterization tile prioritization.

Goals

- Allow Blink to use knowledge of the viewport to reduce the work it does when painting.
- Reduce other work that can benefit from knowledge of the viewport:
    - Fetching resources (e.g. images) that don't affect the extended viewport can be postponed.
    - Animations (e.g. animated GIFs) outside the extended viewport don't need to be ticked.
- Get rid of main-thread viewport computation work (that is, main-thread CalcDrawProps) in cc.
- TODO: Finish this list.

The new approach

The viewport

- What will own the viewport in Blink?
    - A PaintingController associated with one or more documents. This will also be responsible for producing recordings and pushing these to cc.

- The viewport will get updated after layout (and before the compositing update). This means that in the future, we'll be able to avoid creating composited layers for RenderLayers that are entirely outside the expanded viewport.

- Viewport expansion logic will move to Blink, since any optimizations in Blink that use the viewport will really need to use the expanded viewport. Given that the current way to expand (when using impl-side painting) is to simply add a constant number of pixels in every direction, moving the constant to Blink seems like the best initial approach.

    - The viewport expansion logic for non-impl-side painting involves painting more when we're idle. That is, there isn't a single "expanded viewport", but rather high-priority and low-priority regions. Should we support this behavior? (This is required if we want to support non-impl-side painting.) Or should we only support impl-side painting?
        - Let's only support impl-side painting. If we plan on landing any of this before impl-side painting is on everywhere, we'll need a Blink-side flag that enables the new path only when impl-side painting is on.
    - For Android WebView, we'll need to plumb a setting that causes Blink to treat all layers as being entirely within the expanded viewport.

- Each RenderLayer's visible rect, expanded visible rect, and screen space transform will be computed before the compositing update.
    - The logic for computing visible content rects is currently in CalculateDrawProperties, where it is intertwined with the logic for computing target-space visible rects. Since we don't want to expose the concept of render surfaces to Blink, we'll need to untangle some of the logic in CalculateDrawProperties.
    - The logic for this will live in PaintingController.
        - "Painting" may be a confusing name.
    - Scales used for high-DPI and for low-res rasterization will need to plumbed to Blink (so that they are taken into account when recording).
    - When computing visible rects used for determining layers that need to be promoted because of overlap, we will use inflated bounds for layers with CSS transform animations and layers that scroll on the compositor thread.
    - Computing these rects before the compositing update will allow this information to be used when determining which layers need to be promoted.

Invalidation and recording

- Dirty rects will be maintained by Blink in each GraphicsLayer.
    - Invalidations will be generated the same way, but rather than being passed on to WebLayer in GraphicsLayer::setNeedsDisplay and GraphicsLayer::setNeedsDisplayInRect, they will be maintained locally by GraphicsLayer.
- Recording will happen during a painting phase in Blink after the compositing update.
    - Blink will submit a list of dirty rects to a new Recorder object.
    - The Recorder will return a list of rects to record and GraphicsContexts to use when recording.
        - The Recorder will be backed by a PicturePile.

Non-Blink consumers of cc

Other consumers of cc (in particular, ui::Compositor and content::CompositorImpl) will also need to maintain their own viewports. Initially, these will continue to use the existing path. Switching them over to maintaining their own viewport will be handled in a later phase.

Other uses of the viewport in cc

The compositor thread will continue to use the viewport as before. There are no immediate plans for changing this.