khushalsagar@, vmpstr@

SET and Viewports

Aug 8, 2022

OVERVIEW

The Web platform has multiple viewport concepts that impact layout or viewport visible content in the following cases:

- **Root Scrollbar**: When the root element on the page has a static scrollbar which changes the area available to the web content.
- **UI Interfaces:** Interfaces like the URL bar or a bottom bar on mobile.
- **On screen keyboard (OSK):** A virtual keyboard which is displayed next to the web content.
- **Pinch Zoom:** The page scale is changed with a pinch-zoom gesture on mobile devices.

Landing Viewport-relative Units provides an excellent summary of these viewport concepts and how they change in the cases above. The aim of this document is to clarify how the root snapshot should be generated in each of these cases and the UA CSS applied for default animations. The latter also includes cases where the viewport size is different in the outgoing and incoming DOM states.

CURRENT STATUS

The root snapshot is sized to the layout viewport and captures the area covered by this viewport. The UA CSS for sizing/positioning the generated pseudo elements is as follows:

```
html::page-transition(root) {
  position: fixed; inset: 0;
}

html::page-transition-container(root) {
  position: absolute;
  top: 0; left: 0; right: 0; bottom: 0;
}
```

```
html::page-transition-outgoing-image(*),
html::page-transition-incoming-image(*) {
  position: absolute;
  inset-block-start:0;
  inline-size: 100%; block-size: auto;
}
```

An implicit assumption in the CSS above is that the layout viewport is not resized during a transition. The rendering when animating between layouts where the layout viewport shrinks, for instance if the incoming DOM induces a root scrollbar but the outgoing DOM does not, is as follows:

- When the DOM switches to the new layout which reduces the layout viewport bounds, the snapshot is immediately scaled to match the new width.
- The new snapshot is sized to the new layout viewport bounds and requires no scaling since it matches the size of its replaced pseudo-element. The animation is then a simple cross-fade between the 2 snapshots.

PROPOSAL

The abrupt change in the size of the root snapshot is visually jarring.

Capture and Animate Layout Viewport Size

This proposal is to retain the behaviour to capture the layout viewport in the snapshot but animate this size change similar to other shared elements with the following CSS:

```
@keyframes page-transition-container-anim-root {
  /* This is the layout viewport bounds for the outgoing DOM. */
  from { width: 110px; height: 110px; }
}
html::page-transition-container(root) {
  position: absolute;
  top: 0; left: 0;

  /* This is the layout viewport bounds for the incoming live DOM.
     It updates whenever the viewport changes midway in the transition. */
  width: 100px; height: 100px;
  animation: page-transition-container-anim-root;
}
```

The existing CSS for `html::page-transition-*-image` scales the snapshot to match the container width as it animates and the height is scaled such that the aspect ratio remains unchanged.

For the pinch-zoom case, the CSS sizes will need to exclude the page-scale factor (similar to device-scale factor) since it is applied later in the stack.

A con of this approach is that for cases like pinch-zoom, we'll be capturing a scaled snapshot which will include a significant area outside the visual viewport. This can be quite wasteful of gfx memory.

Capture Large Viewport

This proposal is to change the root snapshot to be padded to a consistent size which is unaffected by dynamic UI interfaces like omnibox, OS keyboard and scrollbars. This lines up with the CSS large viewport concept except the large viewport value excludes non-overlay scrollbars.

The padded region will depend on the position of the layout viewport with respect to the large viewport.

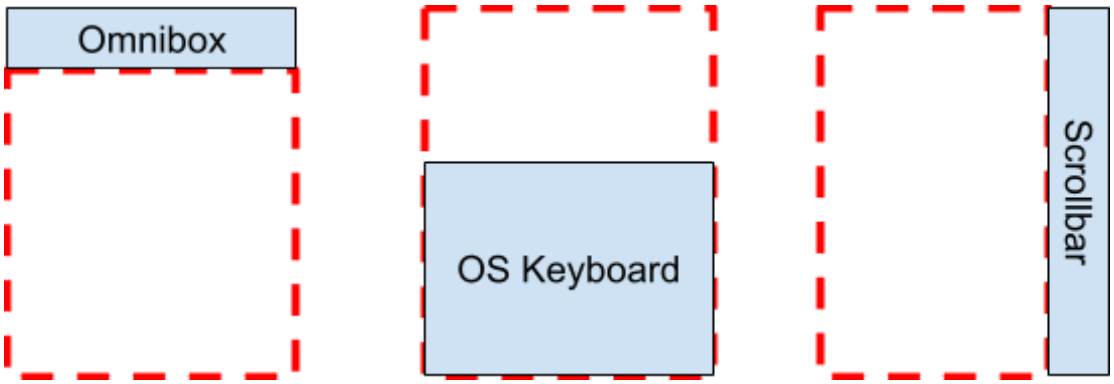

Figure: This assumes the iOS behaviour where OSK doesn't resize the layout viewport. The red dashed box is the layout viewport and the UI interfaces will be the additional padding in the snapshot.

The default CSS applied to position these snapshots is as follows:

```
html::page-transition {
  /* Since fixed position elements are positioned with
     respect to the layout viewport, the top/left values are computed by
     the browser to ensure the origin of the page-transition pseudo-element
     aligns with the origin of the large viewport.
     For example, top here accounts for the omnibox height. */

  position: fixed;
```

```
  top: -10px;
  left: 0px;

  /* These bounds use physical pixel values to include the area covered by
     non-overlay scrollbars. */
  width: 300px;
  height: 1000px;
}

html::page-transition-container(root) {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
}
```

Since the bounds for the container element remain fixed to the large viewport dimensions, no size animations are necessary for the root to root animation. For the cases where this does change (like a window being resized), we could simply finish the transition.

For a root to shared element animation, the keyframes generated animate the width/height of the container pseudo-element from the large viewport bounds to the shared element bounds.

```
@keyframes page-transition-container-anim-root_to_element {
  from {
    width: 300px;
    height: 1000px;
    transform: none;
  }
}
html::page-transition-container(root_to_element) {
  width: 100px;
  height: 100px;
  /* This transform positions the incoming element's border-box with
     respect to the large viewport. */
  transform: translate(100px, 100px);
  animation: page-transition-container-anim-root_to_element 0.25s both;
}
```

Capture and Animate Visual Viewport

TODO: Fill this section

ROUGH NOTES

- Scrollbar switch is immediate. We could fade-in/fade-out for that but won't allow devs to customise it.
- Since user interaction is suppressed during the transition, animation of UI interfaces like omnibox can not be triggered from a user gesture but the browser forces the omnibox to become visible on URL changes for security reasons.