Session storage usage & prerender

Author: jakearchibald@

Public copy

Discussion

The problem

When a same-origin page is prerendered (using `<link rel="prerender">` or `<portal>`), it'll exist in a different browsing context group, and therefore a different process.

It'll have storage access, which includes session storage.

Currently, the only way pages have simultaneous access to session storage is between a page and its same-origin frames. The "storage" event can be used to track changes to session storage between windows, but this seems uncommon in practice; developers often read from session storage on page load, then assume only the current page can make changes to that state.

Bfcache breaks this assumption, as a page could return to active state after another page has modified session storage, but users generally expect that clicking back might appear to put their app in a previous state, and refresh the page to correct issues like this.

Prerendering creates a trickier situation where two pages would share session storage *at the same time*, similar to iframes, but the result will impact forward navigation to previously unseen pages. Also Chrome and Firefox engineers agree that sharing session storage simultaneously across processes would be an implementation challenge.

Options

We have a few options:

A. **Shared:** Make it work similar to iframes by sharing session storage across processes.
B. **Cloned:** Clone session storage for the new prerender context, as we currently do with auxiliary browsing contexts.
C. **Cloned and swapped:** As above, but swap back to the main session storage when the prerender becomes the top-level active document.
D. **Empty:** Associate the prerender context with a fresh empty session storage.
E. **Empty and swapped:** As above, but swap back to the main session storage when the prerender becomes the top-level active document.

Swap event: The swapping solutions could dispatch an event once the session storage has swapped. Maybe the "pageshow" event could be a fit there, since it'd cover bfcache, preprender, and portals.

Review of options against use-cases

I asked folks on Twitter what they use session storage for. Here's a review of the options above in relation to the use-cases folks gave me.

Impact of out-of-sync state

This happens when:

1. Main page is in state 1
2. The prerendered page renders using state 1
3. Actions in the main page put it in state 2
4. The prerendered page activates
5. State 1 appears to have returned

This could result in:

- Inconvenient: State of expanded/collapsed menu items/selected tabs in tab UI reverting to some previous state
- Inconvenient: Stale caches.
- Inconvenient but not user-facing: Incorrect analytics
- **Potentially very inconvenient:** Show-only-once/show-until-dismissed UI reappearing intermittently (typically banners or modals)
- **Serious issue:** user may have logged out in the main page, then appear to be logged back in again when the prerendered page activates.
- **Serious issue:** Items disappear/reappear from shopping cart

Impact of emptied state

This could result in:

- Inconvenient: Expanded/collapsed menu items/selected tabs in tab UI reverting to initial state
- Inconvenient: Missing caches.
- Inconvenient but not user-facing: Incorrect analytics
- **Potentially very inconvenient:** Show-only-once/show-until-dismissed UI reappearing
- **Serious issue:** User logged out
- **Serious issue:** Items disappear from shopping cart

A. Shared

Pages that read session storage once during load (which is common) will encounter out-of-sync state. It'll be racey due the timing of the parent page updating state and the prerendered page reading it.

If the pages react to changes in storage via the "storage" event (which is rare), things should work pretty well, although there may be some races. For instance, the prerendered page could pick up a change in auth state, and fetch relevant data.

A page refresh would correct any out-of-sync issues, but requiring a refresh kinda defeats the point of prerender.

B. Cloned

Pages that read session storage once during load (which is common) will encounter out-of-sync state. There'll be some races due to the timing of the clone and the parent page updating state.

The "storage" event doesn't help in this case. A page refresh wouldn't correct the out-of-sync issues.

Navigating back would result in a switch of session storage, putting the user in a different state, which a refresh wouldn't correct.

C. Cloned and swapped

Pages that read session storage once during load (which is common) will encounter out-of-sync state. There'll be some races due to the timing of the clone and the parent page updating state.

The page could resolve state issues using the swap event. However, since this happens on activation, it might mean some data fetches are delayed, which reduces the effectiveness of prerendering. For instance, the prerendered page could pick up a change in auth state, and fetch relevant user data, but this wouldn't happen until activation time.

This delay in receiving updated state would be more visible with portals.

Pages built assuming only the page can change session storage might hit bugs due to the swap in storage, where in-memory state gets out of sync with session storage.

D. Empty

Pages will encounter emptied state.

The “storage” event doesn’t help in this case. A page refresh wouldn’t correct the out-of-sync issues.

Navigating back would result in a switch of session storage, putting the user in an out-of-sync state, which a refresh won’t correct.

On the up-side, there are no race conditions.

E. Empty and swapped

Pages will encounter emptied state.

The page could resolve state issues using the swap event. However, since this happens on activation, it might mean some data fetches are delayed, which reduces the effectiveness of prerendering. For instance, the prerendered page could pick up a change in auth state, and fetch relevant data, but only at activation time.

This delay in receiving any session state would be more visible with portals.

Pages built assuming only the page can change session storage might hit bugs due to the swap in storage, where in-memory state gets out of sync with session storage.

Verdict

No matter what we do, we’re likely to break developer expectations with session storage. However, since prerender is opt-in, it might be ok to expect a little extra effort to overcome particular issues.

If we can’t do “Shared”, then “Cloned and swapped” seems like the next best choice, as it provides a reasonable starting point for session storage, and updates at activation time.

Session storage use-cases

My analysis of each use-case. I used this to come up with the above recommendation, and I've left it here for evidence/completion. It isn't essential reading 😁

Restoring non-essential UI-state

Such as:

- Expand/collapse state of menu items
- Selected tab in tab UI
- Search term used to get to page
- Scroll position

…between navigation & reloads. It feels like a good history API would be a better fit here.

A. **Shared:** Works if UI reacts to storage events, otherwise UI changes made after the prerender loads will be lost.
B. **Cloned:** UI changes made after the prerender loads will be lost.
C. **Cloned and swapped:** Works if UI reacts to swap events, otherwise UI changes made after the prerender loads will be lost.
D. **Empty:** All UI changes lost.
E. **Empty and swapped:** Works if UI reacts to swap events, otherwise UI changes made after the prerender loads will be lost.

The failures feel like inconveniences.

Tweets:

- https://twitter.com/sirlantis/status/1356342521962618883
- https://twitter.com/_timothee/status/1356285679156617217
- https://twitter.com/reyronald/status/1356309603210956801

- https://twitter.com/cullylarson/status/1356348880812613632
- https://twitter.com/rkaw92/status/1356308698335993856
- https://twitter.com/denno020/status/1357576528939286530
- https://twitter.com/munawwarfiroz/status/1356307851204034561
- https://twitter.com/nod_/status/1356293356406665216
- https://twitter.com/passle_/status/1356526333602516997
- https://twitter.com/oriSomething/status/1356280078032842759
- https://twitter.com/lagerone/status/1356531239994462208

Show-once-per-session UI

Show a marketing/legal information once per session, but only once.

A. **Shared:** Works if UI reacts to storage events, otherwise UI may appear multiple times.
B. **Cloned:** Race condition. UI may appear multiple times.
C. **Cloned and swapped:** Works if UI reacts to swap events, otherwise UI may appear multiple times due to race conditions.
D. **Empty:** UI will appear multiple times.
E. **Empty and swapped:** Works if UI reacts to swap events, otherwise UI will appear multiple times.

Depending on the intrusiveness of the UI, failures could range from inconvenient to very annoying.

Tweets:

- https://twitter.com/Indiequest1/status/1356357421699096576
- https://twitter.com/socalleddom/status/1357083819187265540
- https://twitter.com/oiva/status/1356280650559610886
- https://twitter.com/AhoyLemon/status/1356279949297135620
- https://twitter.com/Indiequest1/status/1356357421699096576

- https://twitter.com/DillonHeadley/status/1356600299616686080
- https://twitter.com/agronmurtezi/status/1356707086550376448

Storing important app state

Such as:

- Shopping cart contents - for cases where a cart could be different per session
- Authentication tokens - folks seem to want this to be per session to work around GPDR, or for strict security reasons.
- Progress in multi-step form

Issues:

A. **Shared:** Works if the page reacts to storage events, otherwise if state changes in the parent page that change will not be reflected in the prerender. In the worst case, a user may have logged out but appear to be logged back in again.
B. **Cloned:** Works if auth data is present and valid when the prerender is created and continues to be fresh when the page becomes active. Otherwise auth state will be out of sync. In the worst case, a user may have logged out but appear to be logged back in again.
C. **Cloned and swapped:** Works if the page reacts to swap events, otherwise if auth state changes in the parent page that change will not be reflected in the prerender. In the worst case, a user may have logged out but appear to be logged back in again.
D. **Empty:** Auth data lost, user will be logged out in the next page.
E. **Empty and swapped:** Works if the page reacts to swap events, otherwise auth data will be absent/invalid when page activates. In the worst case, the user will appear logged out.

The failures seem pretty severe.

Tweets:

- https://twitter.com/SwiftOneSpeaks/status/1356283621556903936
- https://twitter.com/pselas/status/1356538104354603011
- https://twitter.com/jwktje/status/1356284482580586498
- https://twitter.com/markoilic96/status/1356320253358321669
- https://twitter.com/theluk246/status/1356321056093593601
- https://twitter.com/micha149/status/1356504324273278976
- https://twitter.com/nhardy96/status/1356346750844366850
- https://twitter.com/jshado1/status/1356382130499170307
- https://twitter.com/humansJS/status/1356483017682808832
- https://twitter.com/adrjohnston/status/1356297312428040193
- https://twitter.com/PaulieScanlon/status/1356281617048862720
- https://twitter.com/blue2blond/status/1356282535647604742
- https://twitter.com/kennieOutHere/status/1356325015478792195
- https://twitter.com/Fdecampredon/status/1356344507361595393
- https://twitter.com/trammellwebdev/status/1356418006247686153

Identifying a session for analytics

I'm assuming that session storage will be read just before sending some kind of analytics beacon.

A. **Shared:** Just works.
B. **Cloned:** There's likely to be a race where the main page and prerendered page think they're in different sessions.
C. **Cloned and swapped:** There's likely to be a race where the main page and prerendered page think they're in different sessions, but this will be resolved on activation.
D. **Empty:** The prerendered page will think it's in a different session to the main page, even after activation, which is pretty broken.
E. **Empty and swapped:** The prerendered page will think it's in a different session to the main page, but this will be resolved on activation.

The failures feel like minor inconveniences, and not user-visible.

Tweets:

- https://twitter.com/jkarttunen/status/1356283243537063944
- https://twitter.com/prestomation/status/1356299616388411393
- https://twitter.com/marcpicaud/status/1356286843893985280
- https://twitter.com/brandon_duffany/status/1358444486993596417
- https://twitter.com/krogovoy/status/1356322794452168710

Caching

A. **Shared:** Works if the page reacts to storage events, otherwise could result in cache inefficiency. General races could also result in cache inefficiency.
B. **Cloned:** Race condition when cloning, plus loss of things cached between clone & activation lead to cache inefficiency.
C. **Cloned and swapped:** Race condition when cloning could lead to cache inefficiency. Swap event isn't a lot of help here.
D. **Empty:** Cache gone.
E. **Empty and swapped:** Cache gone during load. Swap event isn't a lot of help here.

Tweets:

- https://twitter.com/crouchy/status/1356317356323205120
- https://twitter.com/MattiasBuelens/status/1356309581383794688
- https://twitter.com/warpech/status/1356332649774796806
- https://twitter.com/jeremenichelli/status/1356282299105554435
- https://twitter.com/derSchepp/status/1356292757401251841

Discounted use cases

I think these use-cases aren't impacted by our changes, or are already broken:

- “To save something when you are offline and later send it to a server as soon as the network returns.”
  https://twitter.com/thradams/status/1356281550237794305
  Session storage seems too easily lost for this to be a good usage. Also cloning creates issues.

- “We have a requirement to "lock" an article when in use using a nonce generated per session. Using session storage allows the nonce to remain alive per tab, but will prevent the nonce from being the same in another tab/session”
  https://twitter.com/sirjamespants/status/1356295731032846339
  Cloning already breaks this use-case

- As a fallback if localstorage is disabled
  https://twitter.com/kddnewton/status/1356307071738765314
  Not an issue here

- State used for page refresh
  https://twitter.com/theoomoregbee/status/1356316323547123712
  https://twitter.com/sennevdb/status/1356311230806454275
  https://twitter.com/basketball_gm/status/1356303310203006977
  https://twitter.com/senorflor/status/1356308832201420801
  https://twitter.com/JeddFenner/status/1356420703273299968
  https://twitter.com/actualcactuar/status/1356676702894702598
  https://twitter.com/ZMYaro/status/1356520976284934145
  If it’s only used for refresh, it isn’t an issue here, since we’re only worried about forward navigations.

- Short-term storage for login-flow
  https://twitter.com/gaforres/status/1356328475901227008
  https://twitter.com/DenisTRUFFAUT/status/1356281317579698178
  We’re unlikely to impact this.