PWA Navigation Capturing

dmurph@chromium.org

2024/09/23
Self-link: https://bit.ly/pwa-navigation-capturing-pres
Doc version: https://bit.ly/pwa-navigation-capturing

Context?

- Installable PWAs exist.
- Native apps exist.
- Users click on links & they can open in an app.
- This doesn’t yet exist for installed PWAs.

Goal: Clicking a link to app will open in that app.

"I installed Adobe Express. When I click on a link to a Adobe Express I want it to open in the app, not in a new browser tab."

- This is not new. Web links on mobile (iOS and Android) can open apps.
- Android handles https scheme capturing for patterns

Shouldn’t this be simple?

Naive: If URL is within app, launch app at that url. This should be simple!

However, when launching a similar implementation on Desktop on CrOS, we ran into many issues….

Shouldn’t this be simple?

First approach: If the url is being navigated to is within app scope, cancel that navigation and launch the app at the url!

Easy.

What if the app wants to customize the behavior a bit? We can have declarative options, easy!

- Only use one window (don’t create a new window every time).
- Just tell me in javascript instead of navigating me.
- etc.

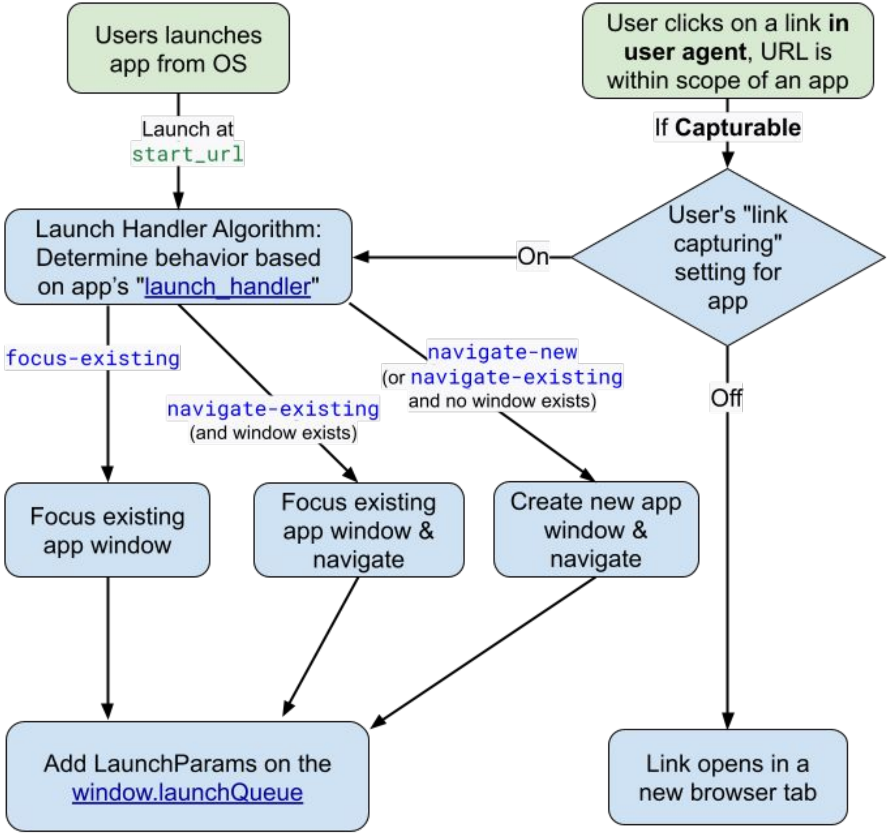
Shouldn’t this be simple?

Apps need to control this behavior

The app declares a launch handler `client_mode` in the manifest:

- `focus-existing`
- `navigate-existing`
- `navigate-new`
- `auto` (default)

Important concept: **Capturable**


Users launches app from OS
Launch at start_url
User clicks on a link in user agent, URL is within scope of an app
If Capturable
User's "link capturing" setting for app
On
Off
Launch Handler Algorithm: Determine behavior based on app's "launch_handler"
focus-existing
navigate-existing (and window exists)
navigate-new (or navigate-existing and no window exists)
Focus existing app window
Focus existing app window & navigate
Create new app window & navigate
Add LaunchParams on the window.launchQueue
Link opens in a new browser tab

Backing up: Definitions

When a url is '**within scope**' of an app, then the app **controls** that url.

Scope is currently defined as a prefix that includes an origin, but there are requests to change this to allow other origins as well as inclusions / exclusions patterns.

A user link click / navigation is **capturable** if it COULD be handled by an app as a launch (and thus handled holistically by an app's `'launch_handler'` behavior).

If a link click/navigation is **capturable**, and an app is installed that **controls** that url, then that navigation is considered **captured.**

Shouldn’t this be simple? - Not ALL navigations

Navigations happen from anchor elements (`<a>`) and `window.open`.

Need to exclude

- navigations from the URL bar
- same-frame navigations
- form submissions
- POST navigations?

What about redirection?

- We can wait for the final `20X` url, and make a decision then.
- But - let’s also try to ignore common redirector patterns like bit.ly so they don’t accidentally get captured before redirection completes…

Shouldn’t this be simple? - Not ALL navigations

What about client-side redirection?

- `window.open(‘’).location = <url>`
- Let’s handle that too - detect `about:blank` redirecting.

Described is basically what is shipped on ChromeOS.

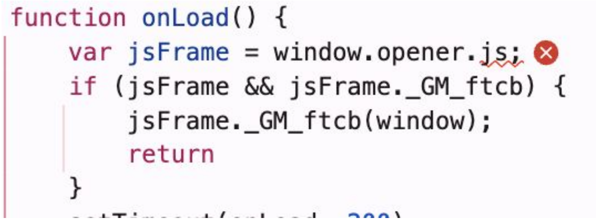
Unfortunately - Problems

- I install Chrome Remote Desktop
- Chrome Remote Desktop authentication opens a popup, which creates a new app window. … but it somehow breaks?

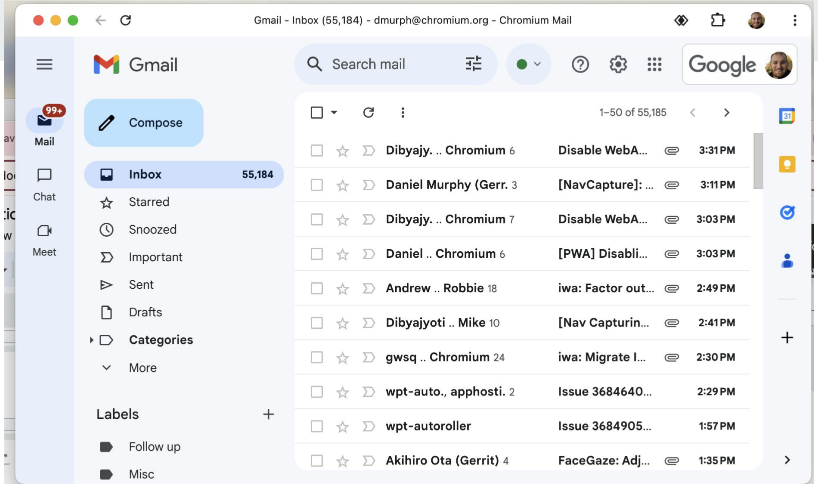
- I install MSFT 365 as an app.
- I create a new ‘word’ document in Microsoft 365 opens a new window but… fails?
- Hint: It was a request with form data…

Unfortunately - Problems

- I install Gmail
- Ctrl-clicking on email from the app
- **Weird:** … it opened in a browser tab, and then in an app window, and then it's blank?
- Even worse, this happens when I'm using gmail from a browser tab too.

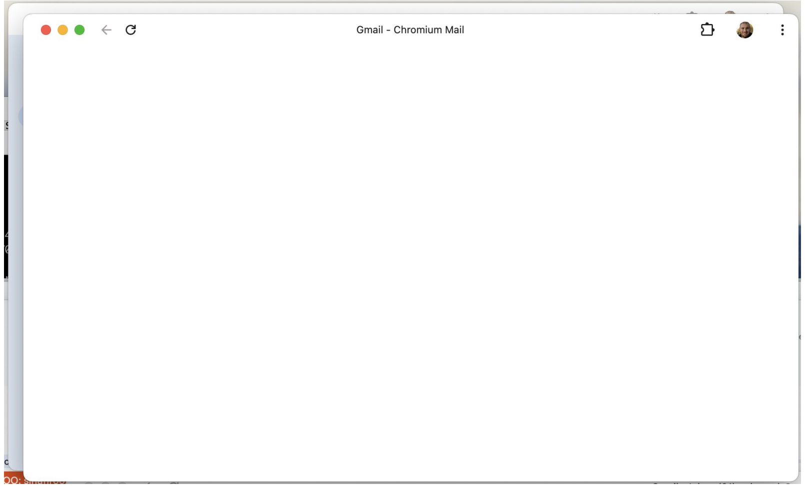


```
function onLoad() {
    var jsFrame = window.opener.js;
    if (jsFrame && jsFrame._GM_ftcb) {
        jsFrame._GM_ftcb(window);
        return
    }
    setTimeout(onLoad, 200)
```



Gmail - Inbox (55,184) - dmurph@chromium.org - Chromium Mail

```
Dibyajy. .. Chromium 6        Disable WebA...    3:31 PM
Daniel Murphy (Gerr. 3        [NavCapture]: ...  3:11 PM
Dibyajy. .. Chromium 7        Disable WebA...    3:03 PM
Daniel .. Chromium 6          [PWA] Disabli...   3:03 PM
Andrew .. Robbie 18           iwa: Factor out... 2:49 PM
Dibyajyoti .. Mike 10         [Nav Capturin...   2:41 PM
gwsq .. Chromium 24           iwa: Migrate I...  2:30 PM
wpt-auto., apphosti. 2        Issue 3684640...   2:29 PM
wpt-autoroller                Issue 3684905...   1:57 PM
Akihiro Ota (Gerrit) 4        FaceGaze: Adj...   1:35 PM
```



Learnings - Auxiliary browsing contexts!

- Gmail & Chrome Remote Desktop were creating these.
- `window.opener` is null
- `let window = window.open(‘url’); window.callJsMethod();`
    - This doesn’t work!

Examples

- Anchor links require **`rel="opener"`**
    - `<a href="www.foo.com/b" target="_blank"` **`rel="opener"`**`>Open link in new window</a>`
- `window.open` creates auxiliary browsing contexts by default (also popups sometimes)
    - `window.open("www.foo.com/a", "_blank")`
    - `window.open("www.foo.com/a")`
    - `window.open("").location = "www.foo.com/a"`

Spec links: creating top-level traversable, auxiliary browsing context

Learnings - Top-level browsing contexts!

There is no `window.opener`, and `window.open(‘url’, ‘noopener’)` returns `null`

Examples

- Anchor links that create new frames imply `noopener` (and thus don’t create aux contexts)
    - `<a href="www.foo.com/b" target="_blank">Open link in new window</a>`
    - `<a href="www.foo.com/c" target="nonExistantFrame">navigate current window</a>`
- `window.open` needs to pass a `"noopener"` argument.
    - `window.open("www.foo.com/a", "_blank", "noopener")`
- Clients API always creates top-level browsing contexts
    - `clients.openWindow(e.notification.data.url).then((windowClient) => (windowClient ? windowClient.focus() : null));`

Spec links: creating top-level traversable, auxiliary browsing context

How are these contexts created?

Links can have properties

- `target`
  - “_blank”
  - “_self”
  - “<name>”
    - (can exist as an embedded frame or not)
- `rel`
  - “noopener”
  - “opener”

Test pages:

- https://sulky-lopsided-bean.glitch.me/
- https://intriguing-veiled-butternut.glitch.me/

How are these contexts created?

When is “Auxiliary” created?

- Anchor link (Left-click only)
  - NOT `rel=’noopener’`
  - `rel=‘opener’`
  - `target=<non-existant-frame>`
- Window.open (all without ‘noopener’)
  - `’opener’` / other combinations also can make a ‘popup’ window.

Generally: We can simply check for `window.opener` to know if a given browsing context is “Auxiliary”.

When is “Top-level browsing context” created?

- Anchor link
  - Left-click
    - NOT `rel=‘opener’`
    - `target=’_blank’`
    - `rel=’noopener’`
      - (target kind of ignored*)
  - Any modified click
- Window.open
  - `‘noopener’` MUST be explicitly specified

Learnings - Navigation properties are important!

We cannot simply ‘abort’ the navigation and do an app launch.

The navigation has a ton of important information form data, impression data, and a bunch of other important properties for privacy & security.

Learnings - Cancelling navigation is wasteful

Navigations occur after creating the frame and web contents, and a lot of work is done (including network stack work).

It also is taking advantage of prerendering.

Destroying all of that to start over is wasteful and likely not needed.

Learnings - User-modified clicks, what do they do?

- Shift click = “open in a new window”
- Middle click = “open in a new background tab”
- Ctrl click = “open in a new background tab”

This generally opens a new top level browsing context - do we always want to capture this?

(calling `window.open` from user-modified click doesn’t force a top-level browsing context)

Learnings - Putting it all together

So what do we want?

- When a site makes a new frame that is tied to the current experience, it should stay in the same experience.
- When a site makes a new frame that is NOT tied to the current experience, then that can be ‘captured’ by an applicable app.
- We should carefully consider what the user may want with modified clicks.
- Redirection needs to somehow work.

Learnings - Previous ‘auxiliary’ concept is perfect!

The ‘auxiliary’ browsing context is exactly the case where we want to maintain ‘container’s for the user.

🎉🎉🎉🎉🎉

Learnings - Putting it all together

There are 4 main categories of support / changes:

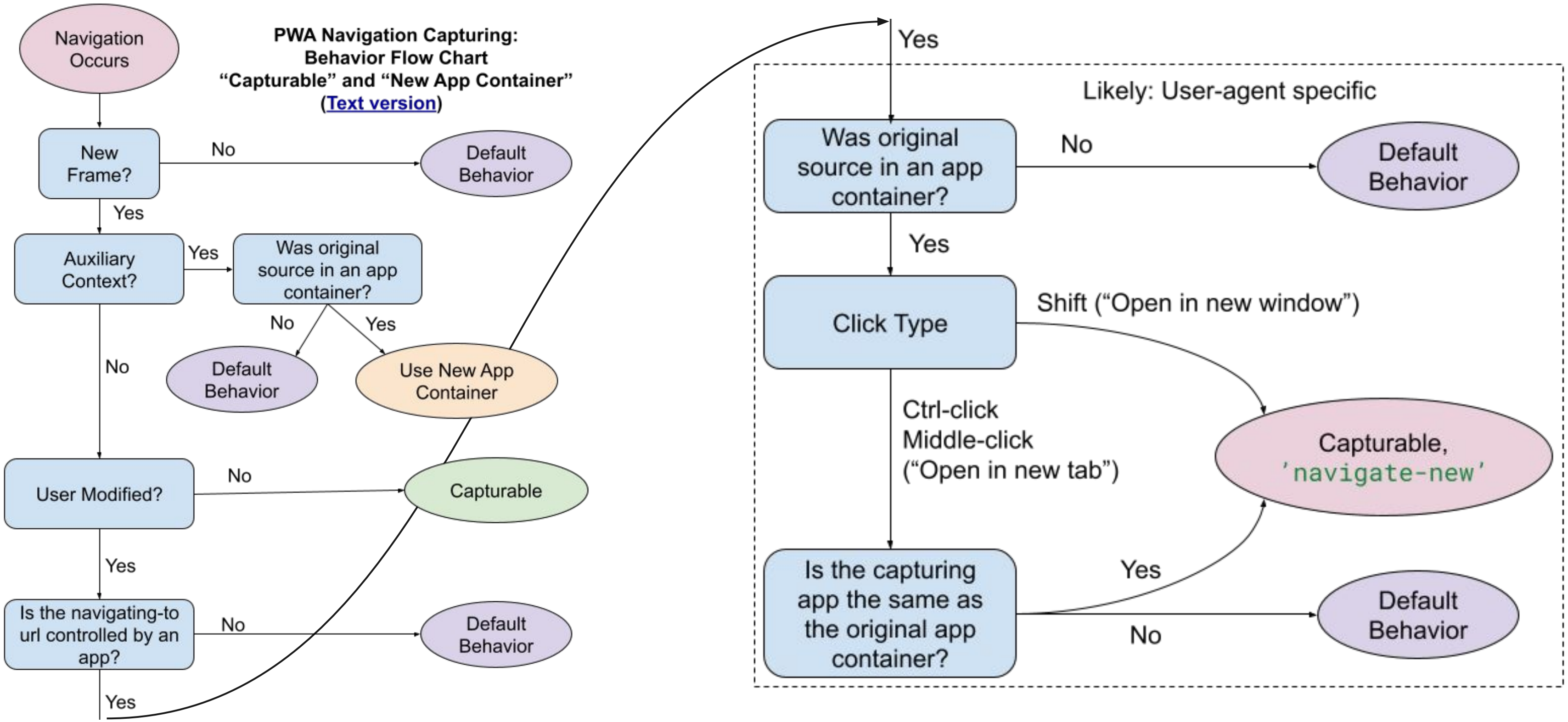
- **Same Browsing Group Fixes**
    - Keep newly created frames in the same container (app or browser) if the site opens an auxiliary browsing context.
- **User Link Capturing**
    - Support navigation capturing to occur when a user clicks on a link controlled by an app, in a way that is compatible with the web platform & controllable by the user.
- **App Windows from Modified Clicks**
    - Define what aux-click behavior should be so that this can open app windows when appropriate.
- **Redirection support**
    - Redirection should result in the same behavior as if the final URL was the one that was first navigated to.

User-modified clicks?

Likely going to be per-user-agent, but Chomium’s take:

- Any user-modified clicks coming from a browser window should never be capturable.
- User-modified clicks from an app that ‘create a new window’
  - should be **capturable** but force the `'navigate-new'` behavior.
  - E.g. Shift-click
- User-modified clicks from an app that ‘create a new tab’ AND are controlled by the same app
  - should be **capturable** but force the `'navigate-new'` behavior.
  - E.g. Ctrl-click, middle-click

Behavior flowchart!



PWA Navigation Capturing:
Behavior Flow Chart
"Capturable" and "New App Container"
(Text version)
Navigation Occurs
New Frame?
No
Default Behavior
Yes
Auxiliary Context?
Yes
Was original source in an app container?
No
Yes
Default Behavior
Use New App Container
No
User Modified?
No
Capturable
Yes
Is the navigating-to url controlled by an app?
No
Default Behavior
Yes
Yes
Likely: User-agent specific
Was original source in an app container?
No
Default Behavior
Yes
Click Type
Shift ("Open in new window")
Capturable, 'navigate-new'
Ctrl-click
Middle-click
("Open in new tab")
Is the capturing app the same as the original app container?
Yes
No
Default Behavior

Table versions

See *https://bit.ly/pwa-navigation-capturing*

Redirection? (server-side)

- “Redirection should result in the same behavior as if the final URL was the one that was first navigated to.”
- The ‘end’ of redirection is know.

Redirection? (server-side)

- Potential edge case - redirectors that can be apps?
  - bit.ly redirector
  - What if bit.ly is installed as an app?
  - We can still ‘do the right thing’ at the end of the redirection.
- “Blinking” into apps or the browser sometimes seems unavoidable.
- Weird case: redirection to ‘navigate-existing’. Cancel navigation? Reparent (breaks client ids and other things)?

Redirection? (client-side)

- “Redirection should result in the same behavior as if the final URL was the one that was first navigated to.”
- MUCH hard with client-side.

Redirection? (client-side)

- Example
  - User clicks on a link that opens a new top-level browsing context to a random site.
  - Site eventually does: `window.location = <url>`
  - .... Is this a redirect that should be captured? Or a same-frame navigation?
- Possible can support:
  - `window.open(‘’).location = <url>`
  - Technically this creates an ‘auxiliary’ browsing context (and the frame is referencable by the parent)
  - MAYBE this can be a special-case that transforms it into a top-level browsing context, and is capturable.
  - **Prior art:** The `Cross-Origin-Opener-Policy` header ‘breaks’ an aux context into a top-level.

Questions?