Permission Element – TPAC 2023 Breakout

Minutes for 13 September 2023 | Permission Element
Github Issue
Explainer
Zoom call link

Presenters:

Penelope McLachlan (google chrome)
mharbach@google.com (google chrome)

Slides

TPAC Permission Control (public)

Present:

- Marian Harbach (Google Chrome)
- Penelope McLachlan (Google Chrome)
- Tara Whalen (Cloudflare)
- Benjamin VanderSloot (Mozilla)
- Alex Christensen (Apple)
- Marcos Cáceres (Apple)
- Theresa O'Connor (Apple, TAG)
- Tim Huang (Mozilla)
- Lucas Haraped (Google Chrome)
- Mihai Cirlanaru (Google Android)
- Chris Fredrickson (Google Chrome)
- Ben Wiser (Google Android)
- Matt Giuca (Google ChromeOS)
- Nicola Tommasi (Google Chrome)
- Thomas Steiner (Google Chrome)
- Tsuyoshi Horo (Google Chrome)
- Ben Kelly (Google Chrome)
- Ari Chivukula (Google Chrome)
- Camille Lamy (Google Chrome)

- Andreas Bovens (Whereby)
- Nicolas Pena Moreno (Google Chrome)
- Christian Dullweber (Google Chrome)
- Gerhard Oosthuizen (Entersekt)
- Tom Van Goethem (Google Chrome)
- Ian Clelland (Google Chrome)
- Ada Rose Cannon (Apple)
- Sameer Tare (Mastercard)
- Rik Cabanier (Meta)
- Xiaohan Wang (Google Chrome)
- Filipa Senra (Google Chrome)
- Kagami Rosylight (Mozilla)
- Yifan Luo (Google Chrome)

Goal

Feedback on a proposal to create a Page Embedded Permission Control (Permission Element)

Scribes

- Matt Giuca (Google ChromeOS)

Slides

TPAC Permission Control (public)

Notes

- Permissions prompts are triggered by either a request to use a capability, or the Permissions API. Shows a UI to the user, asking if the user consents to that capability.
- Problems:
    - User may lack contextual awareness of *why* the permission is being requested (a good site avoids this by making the request contextual, but a bad site will just do it on load or out of context).
        - The “bad” case here may be because the developer doesn’t understand how to do it, or could be a dark pattern, hoping to catch the frustrated user into simply accepting.
    - User flow can be interrupted by permission prompts
    - Permission requests are often outside the user’s attention. Often overlooked during design phase since it’s part of the browser UI, not site UI. Different teams aren’t communicating under less-than-ideal circumstances. (e.g. the JavaScript

team calls the permission request method, and aren't communicating with the UX design team.)
    - Recovering from a blocked decision is hard.
        - Partly a good thing, because we should respect the user decision. But in some cases this is a bad thing as users may change their mind.
        - E.g. In Covid users may have blocked a video chat app in the past not understanding what it's for, then those apps became crucial and it was hard for users to understand how to recover from their past decision.
- Example (screenshot shown) of the lack of attention - user is looking at the bottom-right corner of a large screen, the prompt is in the top-left and the user may not even see it. If they page nav then they may lose the context of why it's asking.
- Example (screenshot shown) of a recovery problem - site says "Permission is blocked" and tries to provide instructions on how to unblock yourself, but you need to give UA-specific advice and it's hard since there's no way to lead users to it. Recovery rate is very low.
- Chrome has shipped some solutions here:
    - Permission request chip: where the lock icon is normally in the URL bar, it says "Share location?" opening up the permission request. It's less interruptive. But unfortunately it's even more out the way. Grant rates dropped from 20% to <1%.
    - In the end, Chrome didn't ship this UI.
- Thinking about:
    - Automatic permission revocation (due to not being used).
    - One-time grants ("Allow this time" vs "Allow every time").
    - We've seen things like this on native platforms.
- Hide prompts that users don't want, based on user behaviour.
    - Something to do with safe browsing.
    - Chrome Permission Suggestion Service, there is a paper published on it (link in the slide - can someone add the link here?)
    - Shipped already. This is ML-based solution, which does help, but isn't a Nirvana.
- The model underlying all these problems is that it's based on websites trying to use a capability and showing a prompt -> a world where the user gets to choose and there's a very high level of user intent that the user wants to do something.
- **The proposal**: A new Permission Element in HTML that reframes from *developer push* to *user pull*. Have to think a lot about mitigations around clickjacking to avoid the developer tricking the user into clicking it.
    - It is spoofable, but spoofing it doesn't really get you anything.
    - It is polyfillable, which would not give you all the benefits of the real permission element, but could fall back to a traditional permission prompt.
- Difficult for the user to trust any content below the "line of death" (i.e. the separation between browser UI and page content). So we have a secondary UI which displays a scrim over the whole page. This is the actual permission prompt dialog (so simply clicking the permission element does not actually activate the permission, it just displays the UI).

- It is intended to be non-interruptive. It is discoverable and can be placed by the developer in context where it is needed. E.g. if it's a store locator it can be placed in the map area. It can be used to revert a previous decision.

Queue

(Add yourself here)

- Tess (Apple, TAG)
    - I don't hate it 🙂
    - Explainer says you want a parser change. (You would require a parser change to not require an end tag.) That probably isn't going to fly. (This is a small detail.)
    - I think this makes sense for some permissions, but not others. As an example, in a world where we didn't have <input type=file> and there was some "get access to file" permission, I would rather that we add <input type=file> than create a permission. Geolocation maybe better off having an <input type=location> than a permission. Should this just be a "plan B" whereas we should have the declarative option as Plan A?
    - Marian: Agree declarative is better. Thinking is required to make it appropriate for each use case. Other cases might not be a fit.
    - Penny: When you look at the explainer, a lot of it is about thinking about ways it could be abused, how much styling is permitted, how to manage trade-off between element blending into the site look-and-feel vs having global consistency. We have a lot of unanswered questions.
    - Johann (Google): A picker pattern is a better model. I don't think that's in conflict with a permissions element which could then open into a declarative model.
    - Tess: I think that would be weird. Where is the value if you have a permission and then trigger a picker. [I may have misinterpreted this.] If you want to show a picker, you should have an API to show a picker, not show a binary permission prompt.
- Ben Wiser (Google Android WebView)
    - I just wanted to flag that the Web Permissions API is not supported in Android WebView because app developers need permission callbacks (ref). There could be an opportunity here to make it more widely supported.
- Ben VanderSloot (Mozilla)
    - I also don't immediately hate this.
    - One point I want to push back on (in the explainer): a non-positive interaction rate does not mean the permissions UI is "bad". We should not use a positive interaction rate as a metric for good UI. Maybe it means the UI is doing its job.
    - Marian: Agree. This isn't about wanting to increase grant rates. This is about asking at the right time.
    - Ben: Three parties: User, Developer, User Agent. As the UA, we should focus on the user.
    - Johann: 97-99% denies on some of these prompts. Other prompts like camera have much higher grant rates.

    - Marian: Data shows that permission prompts associated with a user gesture have tremendous (threefold) acceptance rates.
    - Ben: Also clickjacking is a problem, it's addressed in the explainer. I know you're thinking about it but it still makes me nervous.
    - Penny: A lot of this is capturing user intent. If the user intends to do a thing then says yes/no, we can get a better rate without just popping a box without user's consent. On clickjacking, those who employ dark patterns are often incredibly creative. Will have to think of strategies in advance.
    - Marian: Working with styling team to work out what is the minimal styling needed to reduce the attack surfaces.
- Ari Chivukula (Google Chrome)
    - I agree the potential for spoofing on the grant side isn't really there, but on the revocation side it might be a little? If the permission was already granted, there might be something in a site pretending to handle the revocation. That said, the risk is debatable since the user must have already approved once and many permission types active use already have non-spoofable signals in the browser.One could also argue this isn't really a new concern either, since sites could do this now, but the user is not yet trained to trust such flows.
    - Penny: Agree there is a risk there. I hope that is mitigated by one-time permission grants becoming more the norm, in which case automatically revoked.
    - Marian: Trying to work on our indicators to be a bit more prominent so users have more of a chance to notice.
- Camille Lamy (Google Chrome)
    - Do we foresee requiring specific XSS mitigations for using this?
    - General concern from security people around the impact of XSS on permissions UI. The impact would be that you gain access to any permissions API that has been granted to the origin. Could we think about requiring specific XSS mitigations to nudge developers in the right direction.
    - Marian: What would you do?
    - Camille: Ask for the website to have a reasonable CSP in order to use this.
    - Penny: If there's a real user benefit to transitioning to this from the permissions API, having additional CSP might impede uptake.
    - Matt Giuca (Google ChromeOS): this proposal is not more powerful than what exists already. It's just a different way to prompt. Why should we need an increased security posture?
    - Camille: might make it easier to trick users into granting. Not sure, but we should consider it.
- Gerhard Oosthuizen (Entersekt)
    - Really promising proposal.
    - I like the controlled pop-up once clicked, the potential explained link in the pop-up as well as the proposed ability to show this as active and disable this later.
    - These permissions seem focused on specific hardware/peripherals such as camera/microphone/location.

    - Have we considered this permission for allowing a site to be trusted (e.g. allow future transactions without a challenge)? Even for the irritating cookie popup we all get when it europe.
    - This would be 'active' after a successful payment challenge for example. Allow this site to collect additional signals to protect from fraud.
    - After successful authentication, being able to ask the site to "remember me" or "trust me".
    - Penny: We'd like to tackle the low-hanging fruit first. Cookie consent has big legal ramifications.
    - Christian Dullweber (Google): We are working on storage access permissions.
- Ian Clelland (Google Chrome)
    - The explainer suggests that this can be used (with appropriate policy delegation) from iframes – does the element work the same way in embedded frames as it does when it is clicked in the top-level frame? Specifically, with respect to the site URL which is shown to the user, but maybe there are other differences. What is the user's perception of what's happening in this case.
    - Penny: Haven't explored any differences in the UI when it's in an embedded context. I would imagine that the scrim would still take over the entire page.
    - Marian: We are essentially moving the prompt UI from the top corner and moving it into the centre of the screen. Haven't explored any differences for iframes.
    - Matt Giuca: Let's clarify that moving the prompt from the top corner into the centre, not actually removing a step of trusted browser prompting.
- Ben VanderSloot (Mozilla) (if there is time :))
    - Secondary prompt is just the prompt again, but with entirely different UI than people are accustomed to. What is the delta over a button -> onclick -> permission prompt that you can do now? Just sliding it into page content?
    - Marian: Likely higher that there is user intent. Also that we might remove the block button, and any click outside the prompt dialog just dismisses the dialog. No way to perma-block (dismiss by clicking out of the prompt area).
      [This raised some eyebrows. Further discussion.]
    - The dialog would have "Allow once", "Allow forever" and "Dismiss", not "Perma block". Marian: No need to perma-block since the prompt can't be triggered annoyingly by the site, only by the user clicking a toggle.
    - Ben: You can change your permission UI whenever you want by a JS callback on a trusted element. What is the difference of requiring a user gesture and just letting the developer customize it?
    - Marian: The main difference is that the text in the button is not controlled by the site.
    - Tess: Sites like to customize buttons. You're going to have to allow the site to make it look however it wants using appearance: none. So what's the difference in the appearance: none case. [MG: Sorry I don't know what "appearance: none" is and if I transcribed that correctly…]

    - Penny: We need to capture user intent in order to show that level of intrusive UI. So we need basic control over the experience, can't fully customize the button's appearance.
    - Tess: You're going to have bad adoption if you make it much harder to use this than to just make your own UI.
    - Marian: That's one of the main things we're exploring in the Origin Trial, how much control over the appearance we can get away with.
    - Penny: Early adopters in the OT, looking for them for feedback about what they can live with. The fewer changes we allow, the better, but we're going to have to allow some, and what is that minimal set?
- Ada Rose Cannon (Apple)
    - WebXR generates a permission request when a WebXR session is asked for, could this be used to pre-collect permission for WebXR with a particular session configuration.
    - Additionally could the permission change event count as a user activation to launch into a WebXR session with a single click.
    - Penny: I'm not fully versed in the risks of entering XR mode. (Presume lots of fingerprinting info and things that could be used against the user's interests.) I'd need to give it more thought. But I guess there is precedent for pre-granting permission, but it isn't something we'd historically wanted. E.g. sites asking for location in the user journey before it really needs location.
    - Ada: Probably makes more sense for it to be a system-provided "enter XR" button that only activates when going into that mode.
- Rik Cabanier (Meta)
    - How does this work with multiple permission requests? How much screen estate is used? Will there be a full screen?
    - I think it might be annoying for sites if it has to go into a full screen permission prompt.
    - Penny: Cam+Mic is an example of a joint permission request today. Not sure if we want to explore more fully grouped permissions as part of this (something we've thought about, but it might be putting too much into a single proposal to do it all at the same time). It would be great to model out the cases. The XR case is an interesting instance where you need a lot of permissions simultaneously. It might be good to understand this from a use case perspective, are there discrete groups of permissions we want simultaneous access (as opposed to allowing any arbitrary grouping).
    - Marian: We don't see a lot of simultaneous permissions asks today because permissions are mostly Cam+Mic and Geo.
- Marcos Cáceres (Apple)
    - All except for Geolocation (and maybe Mic) requires a user gesture for a permission prompt. So with that exception, it seems that you can already do all of this just using existing buttons, JS. To get adoption, you'd want new things to be forced to use this.

    - Tess: Are you envisioning a world where the permissions API is no longer available?
    - Marcos: Or just that new APIs have to go through this.
    - Penny/Marian: At the moment just trying to prove that the core idea is viable. But maybe in the future we can move to this model capability-by-capability. It feels like this could be an end state.
    - Marcos: We have a bug open across all browsers to make geolocation require a user gesture.
    - Marian: User gesture only gets you so far. UX research shows that it helps people understand what this is about, but doesn't really help with putting things in context.
    - Johann: Anything that shows a prompt should require a user gesture.
    - Penny: Biggest concern about moving to that today is thinking about all the ways that would break the web that I can't think of.
- Sameer Tare (Mastercard)
    - Can the permission tag be used by an i-frame embedded in the first party page? Coming from Web Payments side of the house, there are payment flows where the bank had to embed content through an i-frame within the merchant webpage to facilitate cardholder authentication. The said i-frame may require permissions to verify certain characteristics of the user.
    - Andreas Bovens (Whereby): this is also of interest to Whereby, as our (mostly telehealth) customers include our video call UI in their pages through an iframe embed.
    - Penny: Current thinking is "yes", this would work in an iframe. But would need permissions policy, valid CSP in place, etc. First-party would have to give that permission to the iframe.
    - Sameer: Things like cookies, etc, is that in scope?
    - Penny: We're looking at whether we've credibly addressed the concerns around clickjacking. Reducing permission decision regret.
    - Sameer: Are you looking at payments? Extra clicks = more friction, which leads to cancellations. Is part of your trial how payments gets impacted.
    - Penny: Payments isn't in the initial set of origin trial.
    - Marcos: (As the editor of the Payments spec) I don't see how this would be applicable.
- Ian Clelland (Google):
    - Iframe permissions policy requirement is already present. CSP / X-Frame-Options is new. Are you concerned about requiring headers?
    - Penny: When a permission is requested in the page, I would hope it isn't adding excessive friction. But that is an assumption on my part, I haven't looked at the data.
- Ben Kelly (Google):
    - Discussion about how the prompt should look (e.g. whether it should have a block button). And a conversation about block rate. We should consider these

together in a success metric, considering a measure of user trust & safety, not just how many times they click accept.
    - Marian: Do users make the decisions for good reasons (as opposed to I just clicked this to make the popup go away). User regret is part of the decision.
    - Penny: Coming back to the UI specifics, this is something where a spec could allow for latitude in the way the UI is presented by the user agent. There could be reasonable variance in the implementation.
    - Tess (Apple): Sometimes I purposely try to figure out how to trigger the permission prompt so that I can click “Never”. It would be disappointing if there’s a button that I can click to tell sites “never” and it doesn’t have that option.
    - Ben: There could be some UI in the browser to say “I never want this site to ask for any permissions at all”. (This already exists today in content settings.)
- Xiaohan Wang (Google)
    - Q: “It is not particularly useful to distinguish between different types of "not granted" states”, should we differentiate b/w “disabled for all sites by default” vs “denied for this particular site”?
    - Marian: If the user somehow configures the site to “never show notifications” then the button could look different.
    - Penny: We probably want the PEPC control to look different and say “you perma-blocked this”.
    - Q: Any concerns that many sites will start to show this element for notifications?
    - Penny: Amongst the considerations would be “element size” (a maximum size for the PEPC element). Hopefully that makes it less attractive than simply using the permissions API to harvest a user gesture to activate the notification permission prompt.
    - Xiaohan: Concern about the button taking up space on the site.
    - Marian: Arguably that’s better, that the site uses their own space up rather than “spending” the user’s attention.
- <AT TIME, CLOSING THE QUEUE>