Web Editing WG TPAC 2023 - 14 Sep 2023 Meeting

Participants

(Please add your name!)

1. Johannes Wilm (Invited Expert)
2. Xiaoqian Wu (W3C)
3. Austin Sullivan (Google Chrome)
4. Sanket Joshi (Microsoft)
5. Olli Pettay (Mozilla)
6. Sean Feng (Mozilla)
7. Edgar Chen (Mozilla)
8. Tomasz Jakut (Invited Expert/CKSource)
9. Dan Clark (Microsoft)
10. Gary Kacmarcik (Google)
11. Mike Jackson (Microsoft)
12. Anupam Snigdha (Microsoft)

Scribes

- _volunteer_

Logistics

<https://www.w3.org/events/meetings/2bdb88b2-64ab-4158-a17a-fe5b88aa897e/>

Agenda

<https://github.com/w3c/editing/issues/438>

Minutes

Topic: Selection handling in Shadow DOM

Recording:
https://w3c.zoom.us/rec/share/LAn-onkXIjowKNOQXKGwT_fJ0p6jO0h8_JYCGeB7Gh0Tyw91Wy1GVe-5vXO53CsO._YDO8BFtJIpA0Wfb
Passcode: +P+kUSA4

Transcript:
https://w3c.zoom.us/rec/share/pxaq0BlbedHxJfGJy4q36rh_RKFu7fxEXTg_v7fzMKyN3JHnGrakcpy3uGf3MYWm.3r407JbjukdguoX0
Passcode: +P+kUSA4

https://github.com/w3c/selection-api/issues/161#issuecomment-1517552762

Firefox bug for ShadowRoot.caretPositionFromPoint,
https://bugzilla.mozilla.org/show_bug.cgi?id=1430307
Firefox bug for getComposedRanges
https://bugzilla.mozilla.org/show_bug.cgi?id=1430308

[cssom-view] ShadowRoot should also have elementFromPoint and elementsFromPoint:
https://github.com/w3c/csswg-drafts/issues/5886

getSelection().addRange should allow range with both boundary points in shadow tree
https://github.com/w3c/selection-api/issues/167

Specify the behavior when DOM mutation occurs across shadow-crossing selection
https://github.com/w3c/selection-api/issues/168

Topic: Delayed Clipboard rendering

- Privacy issue while reading data for web custom types [Delayed Clipboard Rendering] Privacy issue while reading data for web custom types #439

Anupam: Gives brief overview and then presents some issues.
Delayed clipboard rendering - allows delaying the rendering of certain formats until they are needed. A callback is provided with data which is transported to target app.
Issue 439: the custom formats could be used from the source app to figure out into which target apps the content is pasted. Solution: a url of source app is included in the clipboard item so that target app can whitelist urls that it accepts content from. Can decide not to accept from an unknown URL. Works both in native and web target app.
Ryosuke: So work is required in the target app?
Olli: Webkit privacy leak concern is the target app, not the source app. Fix is for other way round.

Austin: We thought more of an option to allow from the outside in general, thereby accepting certain security risks. Pitfall: What if you want to paste between different spreadsheet apps? Large programs may shut out smaller ones. Is against the spirit of proposal.

Ryosuke: Browser needs to handle security, not random apps.
It is a general problem with web custom format, not the delay.
Anupam: Custom formats… this will kind of give a handshake between target and source app.
Austin: Doesn't scale well. What if URLs change?
Johannes: will lock out smaller sites (nexcloud, etc.)
Ryosuke: … or self-hosted blogs
Sanket: Are we saying that …?
Olli: This is really about delay because the source app can figure out which of the provided types are used.
Ryosuke: One could just request all the formats.
Sanket/Austin: That would defy the purpose of the delayed clipboard rendering.
Ryosuke: Leak will occur while the feature exists.
Sanket: Issue occurs for really small formats used by very few sites - meant for use within a specific ecosystem. Excel & co are asking for this feature, but how can we do it without causing these privacy issues.
Austin: Photoshop type can be produced by GIMP or other and it can be consumed by others. The privacy concern is that the source app can see which type is rendered, but it could also be another program. It's just a strong hint.
Anupam: The source app would make a guess, but not know for sure where the paste happens. There will also be certain security risks with non-custom formats.
Austin: It would be a poor user experience if a target app does not accept the clipboard content from a competing app.
Sanket: With this proposal… is there an alternative mitigation that the browser can put in.
Ryosuke: This proposal looks like over-constraint. We will need to think harder and come up with solution.
Sanket: Are there other APIs where we allow it to be used in eco systems internally?
Ryosuke: I don't think there is anything like that.
Sanket: So is the group saying that we believe browsers should provide a solution that protects user privacy, but we don't know how that would work?
Olli: yes
Anupam: Issue is also when copying from native app (?)
Olli: we will need some more time to think.

- What happens to the clipboard data on Tab/Browser Close? [Delayed Clipboard Rendering] What happens to the clipboard data on Tab/Browser Close? #424

Anupam: What happens to data if browser tab closes? Do we render before closing the tab? We discussed different options. We came up with hybrid of solution 2 and 4. Solution: one built-in format that has to be written to clipboard without delay. If a callback is already running, then we cancel it after timeout. Potentially put empty item into clipboard.
Johannes: Will not this create same issues as other solutions: User will get different results at different times without knowing why?
Anupam: This is only if the site has no beforeunload event handler. This is just fallback for that case. Website can show dialog: "You are losing clipboard formats if you close now…"
Olli: not blocking bf cache in any of the browsers

Anupam: Beforeunload event loader can choose one of 3:

- Show dialog with generic text about losing clipboard formats (text provided by browser).
- Render clipboard formats
- Not do anything

RESOLUTION: Microsoft will prototype this solution. It seems to answer all concerns. Will revisit when prototype is ready.

[Break - Terraza de Adhara, Terraza Selene, 1st Floor]

Topic: execCommand/contentEditable

- Define behavior of pressing enter in inline editing host Define behavior of pressing enter in inline editing host #430

Tomasz: We don't use inlineediting host most. Not a real concern for us, because we have our own model.
Johannes: So are you rolling back all changes and do your own
Tomasz: Yes
Simon: I have not checked, but may not be web compat concern.
Ryosuke: Doesn't seem to be used a lot. So not a web compat concern. So we can pick proposed solution
Simon: Yeah, makes sense
Ryosuke: recently implemented to WPT expected behavior. If webkit doesn't match, it's a bug.
RESOLUTION: Simon will document what happens in the three browser engines and check existing wpt tests. Then pick one of the behaviors.

- RFC: Contenteditable "indent" implementations are not aligned with popular editors RFC: Contenteditable "indent" implementations are not aligned with popular editors #437

Johannes: Note that JS editors don't depend on contenteditable default behavior to follow specific indent behavior.
Simon: seems like there is interop. Could break editors/interop. Wouldn't want to change browsers.
Johannes: Would not break modern JS editors. COuld potentially break some older editors (TinyMCE).
Ryosuke: behavior seems to go against native OS behavior. So we would consider that a bug. But it would be risky changing it now that there is interop.
Simon: I would be willing to do compat research if there is implementation interest.
RESOLUTION: Simon will conduct compat research and report back findings and we figure out next steps from there.

- Autocomplete on 'contenteditable' Autocomplete on 'contenteditable' Elements whatwg/html#9065

Sanket: Edge has usecase where we want to provide browser provided writing assistance. Similarly behavior that Grammarly and sites provide themselves. So we need a way to add a way of turning it off if the web author wants to add something custom. Autocomplete is currently

used in forms, and we need something similar. We need something that can show which kind of autocomplete we want.
Simon: I think it could make sense to add form-fill to contenteditable. It seems like this would be a different thing. It would confuse web authors. Could also apply to input elements/textarea.
Ryosuke: So two features?
Simon: Yes
Johannes: The fields in the form are just that specific fields contain specific information. Here it contains many more fields.
Sanket: Could enable multiple and specify what should go in there as well as have a on/off. We can take another name.
Ryosuke: we have autocapitalize and autocorrect that will fix small spelling errors.
Johannes/Sanket: event should work the same way for EditContext as for contenteditable
Sanket: InsertText or insertReplaementText
Johannes: insertReplacementText or a new value that says it comes from generative AI.
Dan/Sanket: insertReplacementText seems like the right value as it's quite similar.
Simon: On Android? Will it be composition or insertReplacementText?
Johannes: What if you want to insert images in the future?
Sanket: If we use more than text, then we probably need a new input type.
Dan: if we only had composition, we could go with that. But insertReplacementText seems closer semantically.
RESOLUTION: We use insertReplacementText as input type
RESOLUTION: It should work the same on EditContext editing hosts as spellcheck works.
Dan: How about "autosuggestions"
Johannes: Why "auto"? "Auto" seems to be a prefix for things that happen automatically.
Dan: Good points, maybe "suggestions"?
Johannes: "input-suggestions"
Dan: sounds good
RESOLUTION: We bikeshed offline. Examples "input-suggestions", "writing-suggestions", "autosuggest" with values like "on" and "off"
Simon: on/off or true/false
…
Ryosuke: autocorrect/autocapitlaize use on/off. That's status quo. If we want to be consistent, then let's do that.
Simon: Value in consistency.

Topic: Input Events

- Adjustments for EditContext [Adjustments for EditContext input-events#143]

Dan: We could make a PR on Input Events that we will merge once we are certain EditContext goes to REC?
Johannes: We could use generic language in some places
Simon: Better to be specific. When will PR be merged?
Dan: Hope to ship this year. We could say trigger is that it is shipped in one browser.

Simon: No official position yet, but some interest. I'll get back with position.
Ryosuke: No official stand yet.
Johannes: Will chromium ship even if it's only them?
Dan: I think if we hear no opposition then yes.
Johnanes: PR can be land as it will not make browsers that don't implement EditContext uncompliant.

RESOLUTION: Make a PR to Input events that makes it work with EditContext, naming EditContext specifically. Not merge until shipped in at least one browser. Dan will write PR.

Input events level 1

RESOLUTION: We publish level 1 as a note. Continue with level 2 through REC-track.

[Lunch - Restaurant Mosaico, Terraza Mosaico, Terraza de Adhara, 1st Floor]

Topic: UI Events

- Composition event order beforeinput and compositionupdate event order uievents#354
    - RESOLUTION: compositionupdate before beforeinput
    - RESOLUTION: order of composition/input events moves to uievents (Gary makes PR to both specs)
    - RESOLUTION: Add example on beforeinput from paste, Johannes will write
- Specify how compositionend works if the caret has been moved to a different element specify how compositionend works if the caret has been moved to a different element uievents#5
    - This bug has a long and hard to follow conversation.
    - We should create a new tracking bug summarizing the current state and asking:
        - Is this still needed? It was originally requested in 2015.
        - Ask Masayuki for current thoughts on this
    - Noting that editContext (which will address the original problem) is targeting late 2023 for shipping.
        - But no official signals from Moz and Webkit, so it's not a cross-browser solution short-term.
        - Can we get signals from Moz, Webkit?
- Export of compositionstart/end https://github.com/w3c/input-events/pull/138
    - Yes it should be exported. See tracking bug: https://github.com/w3c/uievents/issues/356
    - Also defining terms, see bugs:
        - "active text passage" https://github.com/w3c/uievents/issues/358
        - "composition session" https://github.com/w3c/uievents/issues/357

[Break - Terraza de Adhara, Terraza Selene, 1st Floor]

Topic: EditContext

Dan: EditContext - solves problem of creating text editor. Currently the view and the input are the same elements. There are some tricks that work - such a hidden contenteditable that has to be moved so that it is kind of where the IME candidate window will be. There are issues with collaboration when user A has a composition open while user B is sending in changes in. This leads to different issues. EditContext separate things out - the EditContext will receive the input, the JS has to make the changes to the view (could be DOM element, canvas, etc.) without hacks.

- Secure context for EditContext? Secure context for EditContext? edit-context#11

Dan: Should it only work in SecureContext. In issue I have added link to cases when secure context is used - normally when you give access to more data. That is not the case with EditContext. You get access to the same kind of information. Yes, bullet one applies, you do get access to “confidential data” the user types in, but the same is true for contenteditable. I would not limit it, but I am open to other ideas.
Johannes: What about an intranet site that only uses ip numbers, no certificates?
Dan: I don’t think that would not work if we require secure context.
Ryosuke: We don’t see need for secure context.
Johannes: How about Mozilla?
Edgar: I don’t see need for secure context, will bring it up with Masayuki
RESOLUTION: No need for SecureContet for EditContext. Will revisit in case of new arguments from Mozilla.

- Should TextFormat.underlineStyle be an enum? Should TextFormat.underlineStyle be an enum? edit-context#55

Dan: Is minor issue. They are currently strings. Enums could make more sense.
Ryosuke: enum seems like more appropriate type.
RESOLUTION: Switch to enums.

- Should attachedElements() return elements that aren't in the DOM? Should attachedElements() return elements that aren't in the DOM? edit-context#59

Dan: Issue is that there can be lots of old instances that cannot be garbage collected. I don’t think that is a huge issue. I would suggest to just keep them. That way web authors can specify host elements that not yet are part of the dom while setting up the edit context.
Johannes: Maybe note about garbage issue for JS devs in case someone comes up with development pattern where they have lots of tiny editors and keep them stored in an array. But is minor issue. Much more common that editors would tyr to add element as host before connecting it to the dom and then be confused as to why it’s gone when they add the element to the dom.

Ryosuke: Either way seems to make sense.
Edgar: If we want to go the way to destroy the connection, then we need tonotify the event listener that the element is no longer connected to the editcontext.
Dan: Yes, so let's keep the conection.
Johannes: How about a note?
Dan, yeah we can add a non-normative note about it.
RESOLUTION: Don't remove the element from EditContext when removed from DOM. Add a non-normative note about potential garbage collection issue for web authors.