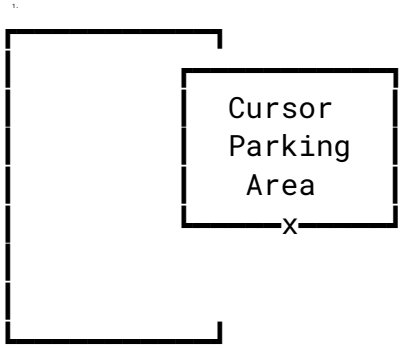
Scoped Custom Elements Registry TPAC 2023 - 13 Sep 2023 Meeting Day 3

Participants

(Please add your name and affiliation!)

- Peter Burns (Google Web Components Team)
- Alan Stearns (Adobe)
- Tom Wilkinson (Google)
- Joey Arhar (Google)
- Justin Fagnani (Google)
- Ryosuke Niwa (Apple)
- Keith Cirkel (GitHub)
- Rob Eisenberg (Blue Spire)
- Oriol Brufau (Igalia)
- Xiaocheng Hu (Google)
- Jesse Jurman (Orthogonal Networks)
- Anne van Kesteren (Apple)

🅿️ Cursor parking



Links

W3C Event: https://www.w3.org/events/meetings/e5cc2d3f-6da6-4751-9417-18b80a524893/

Issue: https://github.com/w3c/tpac2023-breakouts/issues/15
Explainer: https://github.com/WICG/webcomponents/blob/gh-pages/proposals/Scoped-Custom-Element-Registries.md

Minutes

Ryosuke: This meeting is for the scoped custom elements registry proposal.

Ryosuke: Do we want to go over what's in this proposal?

Justin: Yeah. This proposal is a way to get a custom element registry settable per shadow root. The proposal includes some specifics that are involved going through some constraints from implementors, and you can provide a registry to a shadow root when it is created.

Justin: Every shadow root has a fixed registry, either the global one or a local one. From there we provide some APIs for using the registry, e.g. shadowRoot.createElement. Hook into specs for other APIs to find the element registry as defined for that shadow root. There are some other parts that fall out, e.g. the element a registry uses depends on whether it's creating children for its shadow root or for itself.

Justin: We have relatively good agreement, and a list of three open issues, two of which I think we have good agreement on. So how do you [declarative shadow DOM? https://github.com/WICG/webcomponents/issues/914]. Custom element upgrade ordering. Moving elements between shadow roots. I think that describes most of it.

Ryosuke: So I think one question we have is, I think, upgrade… Issue 1001. How does calling define work? Does it upgrade in all effected shadow roots? Because when we do this for the document element registry, we go through and upgrade them in document order.

Justin: Yeah, the order issue is Issue 923. I believe the agreement in the previous meeting was that we'd go in document order. There are other options we could do, but if I recall it was decided that we'd specify document order and let browsers optimize.

Ryosuke: Document order might not be sufficient, might not be a total order. We have to make some sort of decision in what order. It's going to be very expensive, we have to find all of the document trees that might have the element, determine if it uses the affected registry…

Olli: I hope the order wouldn't matter.

Ryosuke: Each custom element registry could be used to construct a shadow tree in a different element. What happens if you create a shadow root in one document, and then adopt the entire tree to a different element? Do we use the ordering of the original document?

Olli: [... some missed things, but definitely “probably this is rare”]

Anne: if a template is on one document, and clone it, then it will be only in the new document.

Ryosuke: But you could also import without cloning.

Anne: People definitely move elements between documents. Is this common though?

Ryosuke: For each document tree, in that cluster we have to traverse the whole thing and find all the shadow roots and upgrade the elements.

Justin: Is this more expensive than what we have to do today with the global custom element registry?

Xiaocheng: We have to do bookkeeping anyways, but I also think it’s just an implementation detail. We could do document order, which is easy.

Anne: You’re speaking to implementors.

Justin: Xiaocheng has also been implementing in Chrome.

Xiaocheng: It’s also mentioned that the same registry can be used across multiple documents. I would ask for a clarification, are we trying to define ordering for interop or are there use cases in mind?

Justin: There is going to be reliance on orderings by user code.

Xiaocheng: I recall a document suggesting that custom element authors shouldn’t design elements that depend on such ordering.

Justin: That may be, but I think we can maybe do something like, we upgrade in document order.

Ryosuke: My point is that you can have a shadow root moved from one document to another. In that case, what order do we use, the order of the resulting document tree?

Justin: If ease of describing the behavior is the point, could the answer to Ryosuke’s question be just to define an order between documents?

Ryosuke: But that would mean that every registry would have to keep track of each document it’s used in. [and something about a top-level document]

Alan: Is this the main issue? The crucial issue?

Anne: This has been a crucial issue for 3, 4, 5 years.

Alan: Time check, we've gone through half an hour on this subject.

Anne: We might have enough detail here for a proposal keeping a set of documents for each registry.

Ryosuke: If order between [...] doesn't matter, then we could use the custom [...]

Justin: Implementors might keep a backlink from registry to document, but we don't want to require it of implementors. One proposal was that each registry keeps a list of shadow roots it was used with and you upgrade them in that order.

Anne: With regard to observability, it could leak [...]

Ryosuke: Maybe the document order is the lesser evil. In the common case you have one document. In the rarer multi-document scenarios you could have an unordered set of documents.

Anne: So you're suggesting no book keeping at all?

Ryosuke: You'd keep the ordered set of documents.

Anne: By doing that you've got to clean it up when a document goes away.

Ryosuke: Yeah, it needs to be a weak reference, not a strong reference.

Anne: You want to check that it's removed correctly. We wouldn't want to remove it when [...]. Would we still want to run registry code if the document is unloaded?

Ryosuke: Do we have a resolution here? Does anyone disagree with using an ordered set of documents?

Anne: And in particularly, use it for [...].

[...]

Ryosuke: You could imagine [implementation details]

Justin: If we're keeping count of the number of shadow roots that use a registry, but you'd have to update that count on every dom operation, like removing a subtree.

Olli: [unclear]

Justin: One extra thing that I think was not clarified in the proposal related is that we're not going to upgrade disconnected trees.

Ryosuke: There's also the question of frameless documents. What do we do if we move a shadow root into a detached document without a browsing context.

Olli: Are there use cases for this scenario?

Justin: I can't think of any. The use cases I know of for multiple documents in this area is using iframes as part of a polyfill for scoped registries.

Ryosuke: One use case is iframes, which can navigate and get detached from the browsing [...].

Olli: [inaudible]

Ryosuke: So if there's no use case for this, my preference is to only upgrade elements with a browsing context. Does anyone disagree with that proposal?

Olli: That sounds good to me, we can change this later if use cases come up.

Ryosuke: All right, sounds like we have a resolution. We will use the ordered set of documents.

[Resolution posted in Issue #923 (Comment)]

Moving on to https://github.com/WICG/webcomponents/issues/1001

Justin: The existing spec has customElements.upgrade for upgrading all custom elements in a subtree. How does this work with a scoped registry?

Ryosuke: [...]

Justin: If you had a tree that had shadow roots, and they use multiple registries across different shadow roots, and you had some elements across those shadow roots and they weren't upgraded… it's hard to construct a case where this matters. If you have a detached tree with multiple shadow roots, and you call customElements.upgrade, the user intent it seems to me is to make the whole subtree live, and to upgrade all elements. I'm trying to think of a case where someone would want option 1, where you'd stop at a shadow root and not upgrade its children

Ryosuke: Could end up with element gets upgraded but its own shadow tree doesn't.

Justin: Element could add callback when upgraded so it could upgrade its own shadowroot… but we probably don't want to do that.

Xiaocheng: What happens if you call upgrade from one registry on a root that's using a different registry?

Justin: Under option 2, it doesn't matter what registry you call upgrade on. It will always upgrade every subtree for every registry. Does anybody think it's a problem to upgrade all shadowroots using any registry?

Olli: We want to do one pass, top down tree walk to upgrade.

Ryosuke: Yeah, we don't want to do multiple passes. Given what we decided earlier, that points towards option 2 I think. We have to pick sensible behavior, but it probably won't matter much what solution.

Anne: So in particular the question is what customElements.upgrade has to do given a node?

Ryosuke: With option 1, when you call upgrade with a registry you only upgrade elements for that registry. With option two, regardless of the registry, we upgrade all elements.

Anne: Isn't the way it works, when you call upgrade it upgrades the entire subtree?

Justin: Yes, but scoped registries introduce an ambiguity.

Anne: What if there isn't an upgrade method on each registry, there's just upgrade on the global registry.

Justin: We'd need a different class for the global registry, but that might be desirable for other reasons.

Ryosuke: [approving comment, not caught by scribe]

Justin: Ok, I can update the proposed spec with this.

Justin: I think there is a question, since Xiaocheng is prototyping in Chrome and that's raising some of these questions. Is anybody else looking to prototype? We've got major customers chomping at the bit for this, and are looking to ship the polyfill to production. We're making changes to try to avoid a smooshgate, but there's major custom interest here

Anne: The resolution we've got here sounds good. I thought we'd already agreed that upgrade was only on the global but I might be misremembering.

[Resolution posted in Issue #1001 (Comment)]

Justin: Going through in document order may have resolved the other open issue, issue 907. Ryosuke could you take a look?

Ryosuke: Hopefully it is resolved

We are out of time!