

How to handle future constraints

What has happened?

The spec currently specifies the use of an IANA registry, and that since a long time: discussed&adopted at the Santa Clara 2011 TPAC.

Discussion popping up from time to time over the years: right way or not?

LC comment (<https://www.w3.org/2006/02/lc-comments-tracker/47318/WD-mediacapture-streams-20150414/3024>): can we remove registry? (Dated April 20)

Consensus poll June 2015 (<http://doodle.com/poll/9v3beu55pgtq4ndh>): no consensus

Dom's mail on requirements (only Harald responded so far) <https://lists.w3.org/Archives/Public/public-media-capture/2015Jul/0016.html>

Recent events show that new constraints will be proposed (antiFlicker)

List of proposed requirements on “registry” and process

- SHOULD prevent duplicates (same constraint, different name)
- MUST prevent same name used for different constraints
- MUST ensure each con. is well enough described to ensure interoperability
- MUST ensure architecture of gUM constraint is maintained
- MUST ensure that definition is in valid WebIDL
- SHOULD ensure that privacy threatening constraints are not introduced
- SHOULD make it easy to find all constraints
- SHOULD be easy to use (so implementers are likely to register exp. con.)
- SHOULD protect implementors from IPR issues
- MUST operate under clear governance
- SHOULD offer strong assurance for long term maintenance
- MUST be easy for newcomers to propose new constraints
- SHOULD be easy to find from initial list

Grouping: requires domain knowledge, easy to find, easy to use, requires IPR clearance, governance/stability

- SHOULD prevent duplicates (same constraint, different name)
- MUST prevent same name used for different constraints
- MUST ensure each con. is well enough described to ensure interoperability
- MUST ensure architecture of gUM constraint is maintained
- MUST ensure that definition is in valid WebIDL
- SHOULD ensure that privacy threatening constraints are not introduced
- SHOULD make it easy to find all constraints
- SHOULD be easy to use (so implementers are likely to register exp. con.)
- SHOULD protect implementors from IPR issues
- MUST operate under clear governance
- SHOULD offer strong assurance for long term maintenance
- MUST be easy for newcomers to propose new constraints
- SHOULD be easy to find from initial list

Reflections

- Many requirements call for domain knowledge (red ones on previous slide)
- We may have conflicting requirements:
 - So easy to use so people register experimental constraints AND
 - Offer IPR protection, make sure two implementations interoperate, maintain architecture, (more)
 - -> Reason to break “experimental” phase from “tried and proven”?
 - Would an IANA registry fill one or both roles?

Reflections

- Somewhat orthogonal to the processe(s) used is the question about where and how it (the process) should be documented (“easy to find”)
 - Part of gUM spec?
 - On a WG Wiki?