Zyte (formerly Scrapinghub):

Extend CSS Selectors Level 4 Support in cssselect

About me

Contact info: github - annbgn

Code contribution to selected organization

pull request for issue #24. Status: awaits code review

Project information

Sub-org: Zyte (formerly Scrapinghub)

Goal: make csselect support CSS Selectors Level 4 as said in issue #108

Abstract: cssselect currently parses CSS3 Selectors and translates them to XPath 1.0 expressions. It is requested to be able to parse CSS4 Selectors too.

Timeline:

I googled up difference in css level 3 and css level 4 selectors, and found out that in level 4 these features were added: “*Added the || column combinator, grid structural selectors, logical combinators, location, time-dimensional, resource state, linguistic and UI pseudo-classes, modifier for ASCII case-sensitive and case-insensitive attribute value selection*.” This may serve as a roadmap, but I suggest fixing known issues in cssselect repo first, as these are things requested by users. Now there are 4 open issues: #51, #66, #67, #76 and I guess every issue will take about a couple of weeks to identify its scope, ask questions to issue authors (optional), code some solution, write or fix tests and wait for pr to be approved. Then I will be free to provide solutions for other differences in css3 and css4 selectors mentioned above and write tests.

May 17 - June 7: community bonding, as said in official timeline

June 7--20: solve #66 issue, write tests. Example `p:has(strong)` -> `descendant-or-self::p[name() = 'strong']`. Reuse code in #96. Support expressions like `dt:has(> dt)` to transform into `descendant-or-self::dt/child::dt`. Be able to parse forgiving-selector-list with its relative selectors and then process selectors with combinator_mapping

June 21 - July 4: solve #67 issue, write tests. Example `div:matches(.active)` -> `descendant-or-self::div[@class and contains(concat(' ', normalize-space(@class), ' '), ' active ')]`. Reuse code in #109.

July 5-11: solve #76 issue, write tests. Example `:nth-child(-n+3 of li.important)` -> `descendant-or-self::*[count(preceding-sibling::li[@class and contains(concat(' ', normalize-space(@class), ' '), ' important ')] ) <= 2]`. Process '`of S`' part and return in `parse_series()`. S is a subselector.

July 12-18: solve #51 issue, write tests. Example `:not(a > b)` -> `descendant-or-self::b[..[not(name() = 'a')]]`. First, while processing the negation decide if its argument is simple or not. In the first case use existing flow, in the latter case parse it so that it matches the example.

July 19 - August 1: support logical combinators. Logical combinations are `:is()`, `:has()`, `:not()` and `:where()`. First three already have some implementations, so I have to support `:where()` by creating a new class with `canonical` and `specificity` methods.

August 2-8: implement `:any-link()`. It is equivalent to `:is(:link, :visited)`, and all of these things are already implemented. It is needed for cssselect to accept several arguments in `:is()`

August 9-15: this time should be taken by unfinished tasks and minor bugfixes if any else documentation updating

August 16-23: submitting code and final evaluations, as said in official timeline

Stretch goals (may be achieved if time permits):

- expand processing of `:matches()` to `:is()` and `:any()`, since both are aliases to `:matches()`
- support 'of S' part in `:nth-of-type` and `:nth-last-of-type`. It was not requested in the issue #76, but the work seems very similar to `:nth-child` (see July 5-11)
- allow `:has()` and `:not()` to be nestable
- support not only >, but also + and ~ in `:not()` (related to July 12-18)

Not part of this project

- || column combinator, why: couldn't find in cssselect anything connected to columns, so the feature itself might become a tip of an iceberg and require implementing a lot of stuff connected to tables. also it is marked at-risk
- grid structural selectors, why: same as above
- time-dimensional, why: not sure if xpath has anything connected to time
- resource state, why: same as above
- linguistic and UI pseudo-classes, why: limited timeframe. However, it looks like it's not impossible to implement, because this point consists of `:dir()` and `:lang()` pseudo classes, whose main difference from `[dir=something]` and `[lang|=something]` is that it matches not only the elements with explicitly set attributes dir and lang, but also elements which have ancestors that have the attribute
- modifier for ASCII case-sensitive and case-insensitive attribute value selection, why: time limit