Selectorgadget

Hadley Wickham

2016-06-16

Selectorgadget is a javascript bookmarklet that allows you to interactively figure out what css selector you need to extract desired components from a page.

Installation

To install it, open this page in your browser, and then drag the following link to your bookmark bar: SelectorGadget.

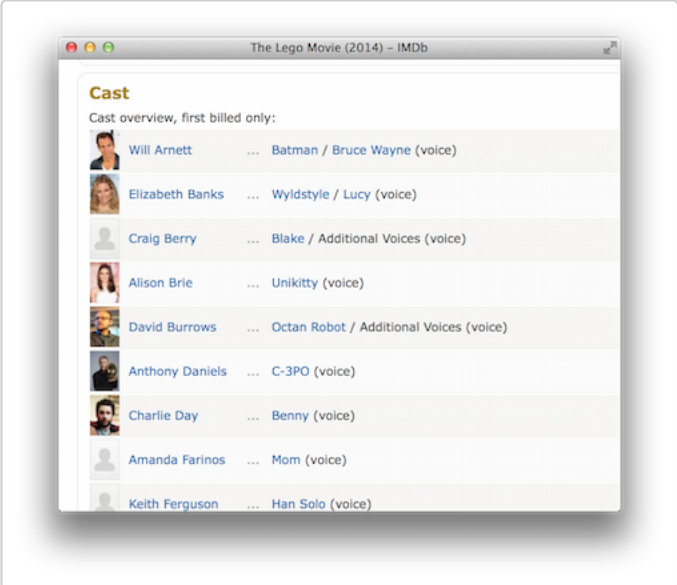
Use

To use it, open the page

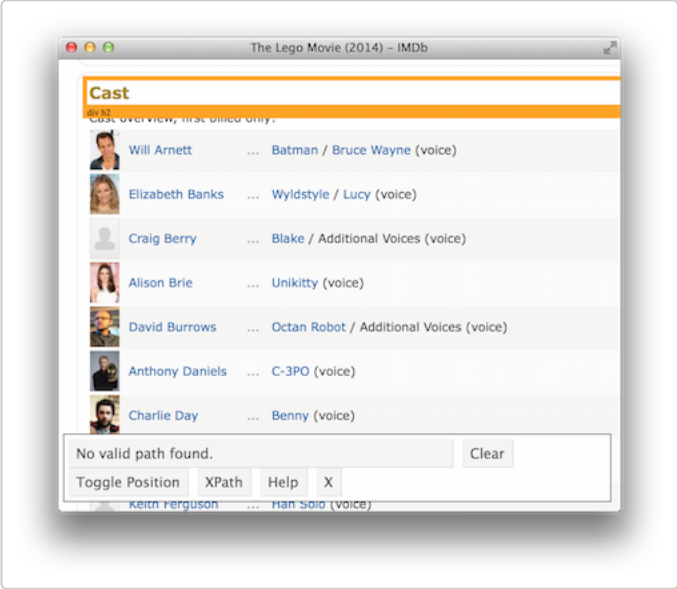
1. Click on the element you want to select. Selectorgadget will make a first guess at what css selector you want. It's likely to be bad since it only has one example to learn from, but it's a start. Elements that match the selector will be highlighted in yellow.
2. Click on elements that shouldn't be selected. They will turn red. Click on elements that *should* be selected. They will turn green.
3. Iterate until only the elements you want are selected. Selectorgadget isn't perfect and sometimes won't be able to find a useful css selector. Sometimes starting from a different element helps.

For example, imagine we want to find the actors listed on an IMDB movie page, e.g. The Lego Movie.

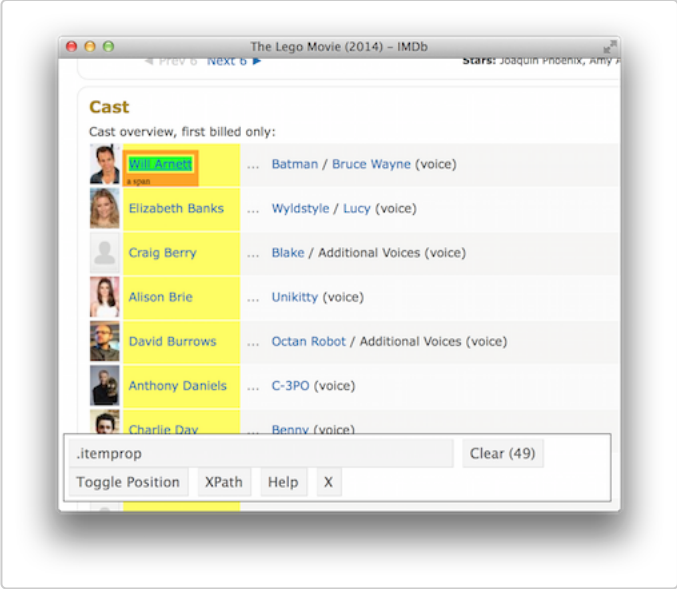
1. Navigate to the page and scroll to the actors list.



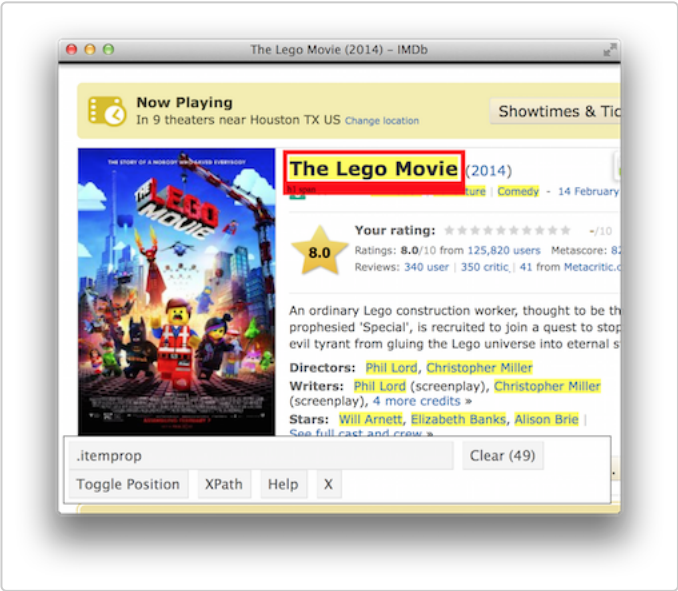
2. Click on the selectorgagdget link in the bookmarks. The selectorgadget console will appear at the bottom of the screen, and element currently under the mouse will be highlighed in orange.

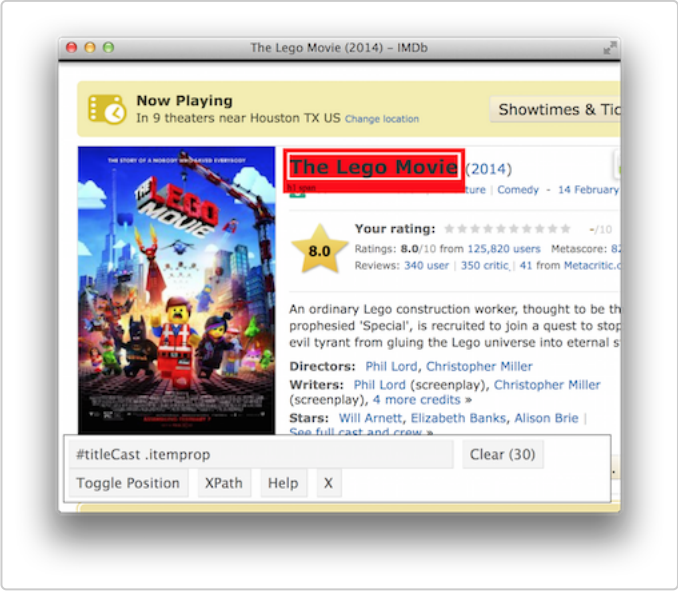


3. Click on the element you want to select (the name of an actor). The element you selected will be highlighted in green. Selectorgadget guesses which css selector you want (`.itemprop` in this case), and highlights all matches in yellow.



4. Scroll around the document to find elements that you don't want to match and click on them. For example, we don't to match the title of the movie, so we click on it and it turns red. The css selector updates to `#titleCast .itemprop`.





Once we've determined the css selector, we can use it in R to extract the values we want:

```
library(rvest)
#> Loading required package: xml2
html <- read_html("http://www.imdb.com/title/tt1490017/")
cast <- html_nodes(html, "#titleCast .itemprop")
length(cast)
#> [1] 30
cast[1:2]
#> {xml_nodeset (2)}
#> [1] <td class="itemprop" itemprop="actor" itemscope="" itemtype="http:// ...
#> [2] <span class="itemprop" itemprop="name">Will Arnett</span>
```

Looking carefully at this output, we see twice as many matches as we expected. That's because we've selected both the table cell and the text inside the cell. We can experiment with selectorgadget to find a better match or look at the html directly.

```
cast <- html_nodes(html, "#titleCast span.itemprop")
length(cast)
#> [1] 15

html_text(cast)
#>  [1] "Will Arnett"     "Elizabeth Banks" "Craig Berry"
```

```
#>  [4] "Alison Brie"     "David Burrows"   "Anthony Daniels"
#>  [7] "Charlie Day"     "Amanda Farinos"  "Keith Ferguson"
#> [10] "Will Ferrell"    "Will Forte"      "Dave Franco"
#> [13] "Morgan Freeman"  "Todd Hansen"     "Jonah Hill"
```