DOM Selectors The Complete Web Developer in 2018

The Complete Web Developer in 2018
Zero to Mastery
Andrei Neagoie
Lecture Notes by Stephanie

DOM Selectors
----

getElementsByTagName	less powerful than querySelectors
getElementsByClassName	
getElementById	
querySelector	similar to CSS, recommended
querySelectorAll	
getAttribute	
setAttribute	

##Changing Styles

style.{property}	ok - messes up separation of concerns
className	best
classList	best

classList.add
classList.remove
classList.toggle

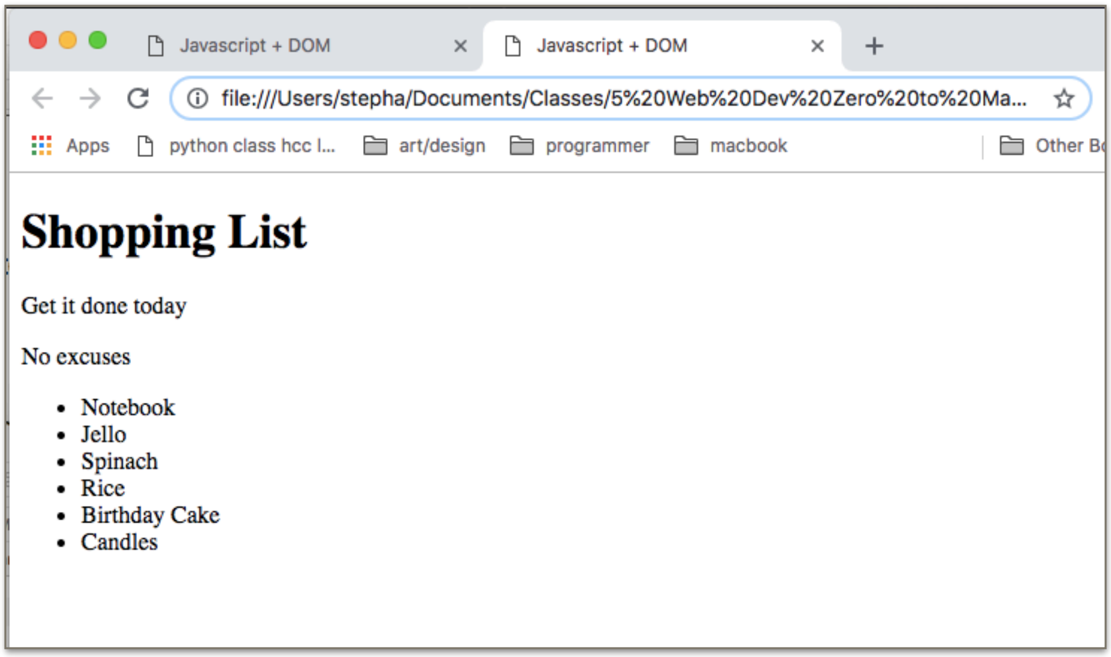
##Bonus

innerHTML	DANGEROUS

parentElement
children

##It is important to CACHE selectors in variables

Lets look at website with this HTML:



Shopping List

Get it done today

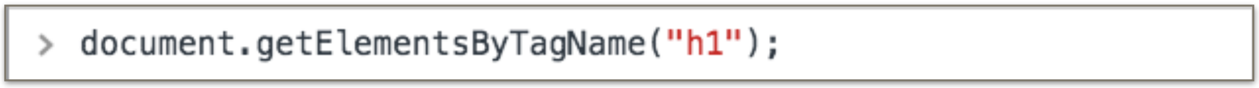
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

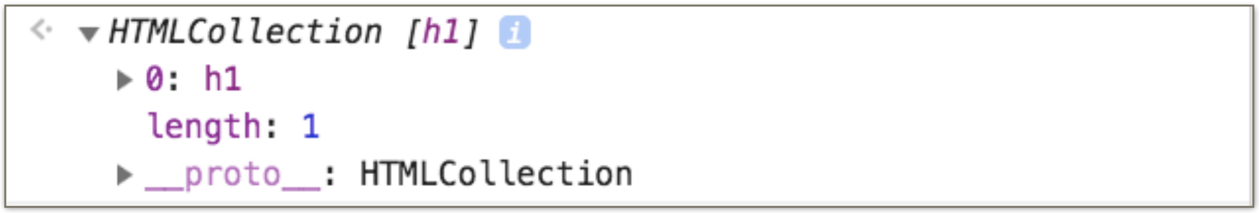
```
<!DOCTYPE html>
<html>
<head>
	<title>Javascript + DOM</title>
	<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
	<h1>Shopping List</h1>
	<p id="first">Get it done today</p>
	<p class = "second">No excuses</p>
	<ul>
		<li random="23">Notebook</li>
		<li>Jello</li>
		<li>Spinach</li>
		<li>Rice</li>
		<li>Birthday Cake</li>
		<li>Candles</li>
	</ul>
</body>
</html>
```

- - - - — - - - - - - DOM Selectors - - - - — - - - - - -

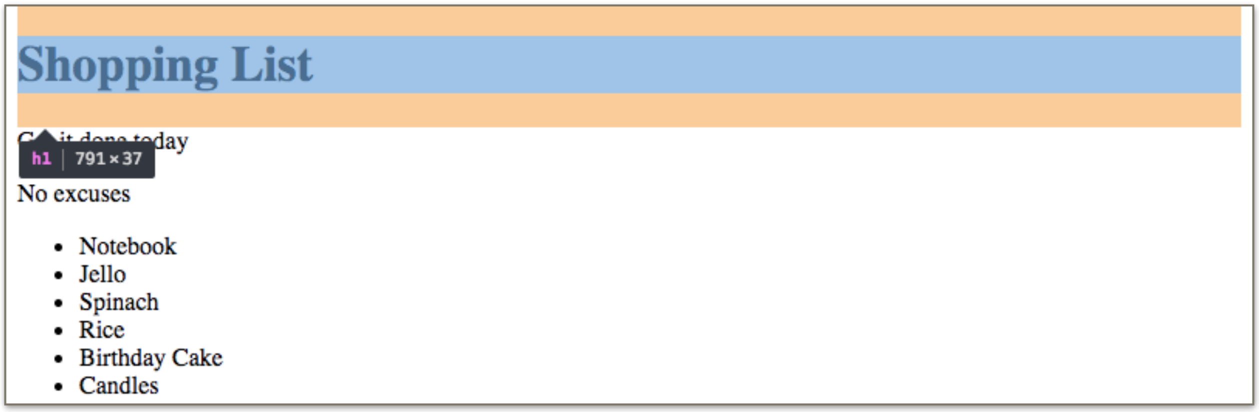
getElementsByTagName



```
> document.getElementsByTagName("h1");
```



```
<· ▼HTMLCollection [h1] i
     ▶0: h1
      length: 1
     ▶__proto__: HTMLCollection
```



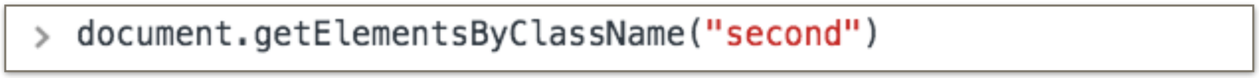
Shopping List

Get it done today

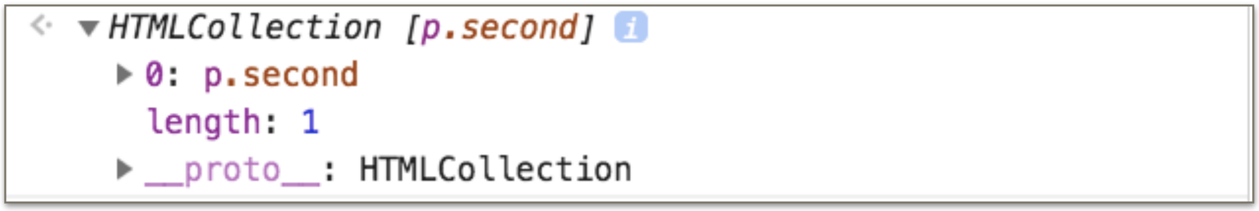
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

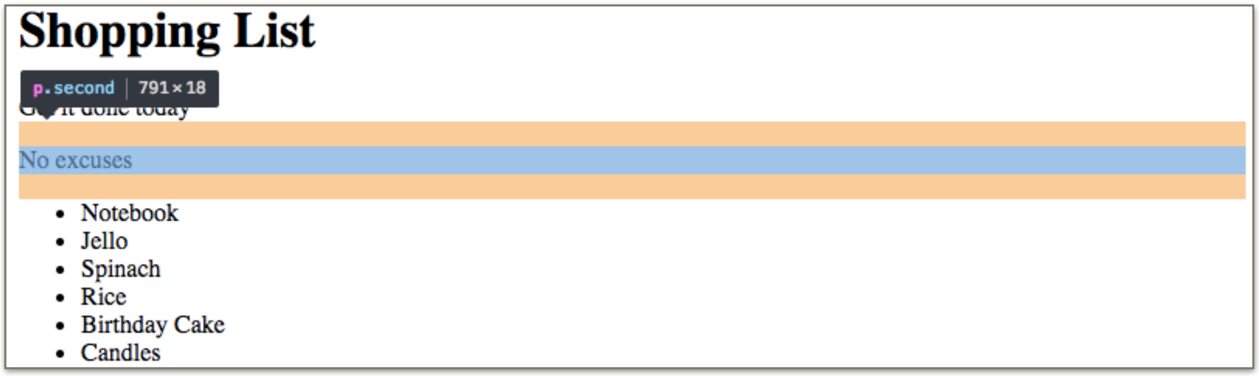
getElementsByClassName



```
> document.getElementsByClassName("second")
```



```
<· ▼HTMLCollection [p.second] i
     ▶0: p.second
      length: 1
     ▶__proto__: HTMLCollection
```



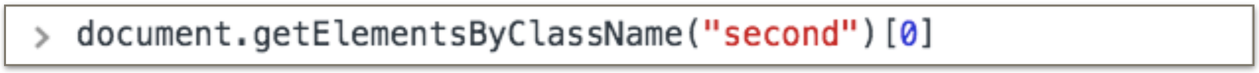
Shopping List

Get it done today

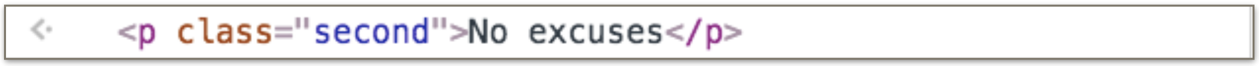
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

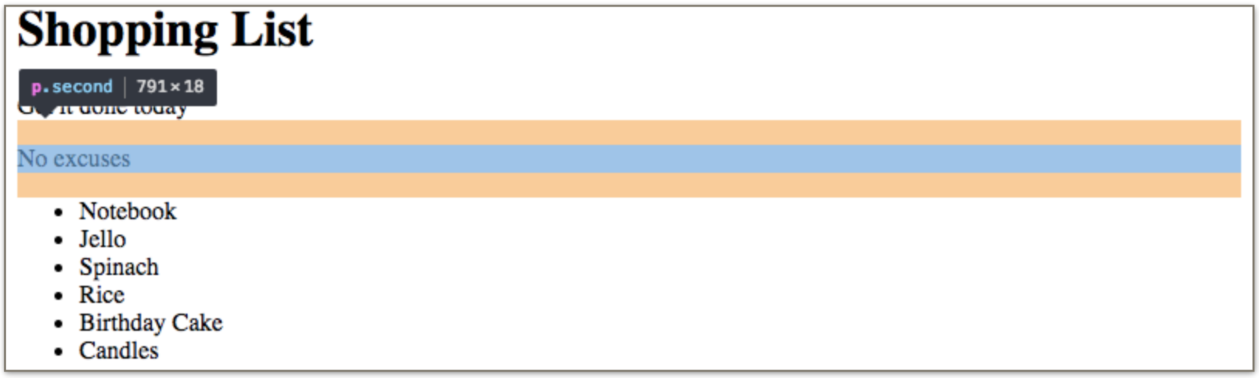
item at index [0] of class, getElementsByClassName



```
> document.getElementsByClassName("second")[0]
```



```
<·  <p class="second">No excuses</p>
```



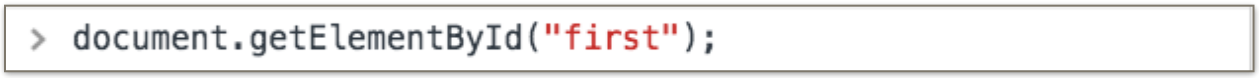
Shopping List

Get it done today

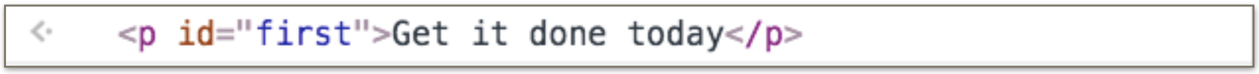
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

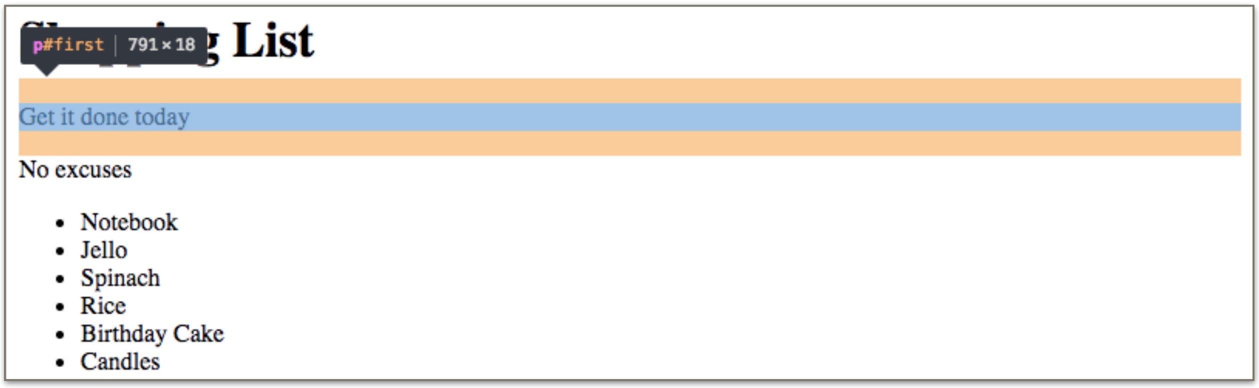
getElementById



```
> document.getElementById("first");
```



```
<·  <p id="first">Get it done today</p>
```



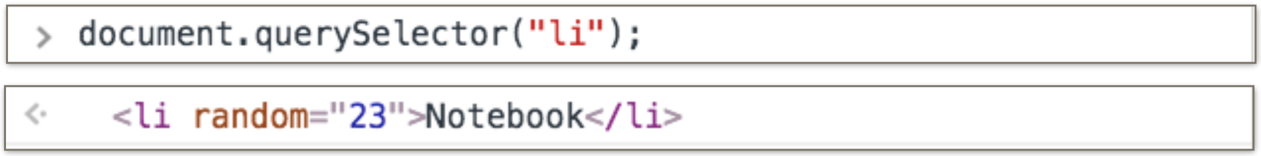
Shopping List

Get it done today

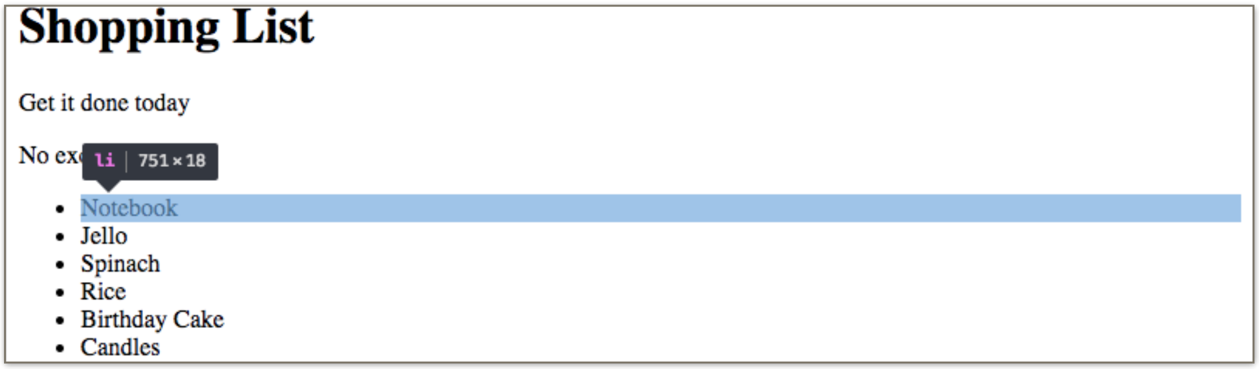
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

querySelector - only selects first element



```
> document.querySelector("li");
<·   <li random="23">Notebook</li>
```



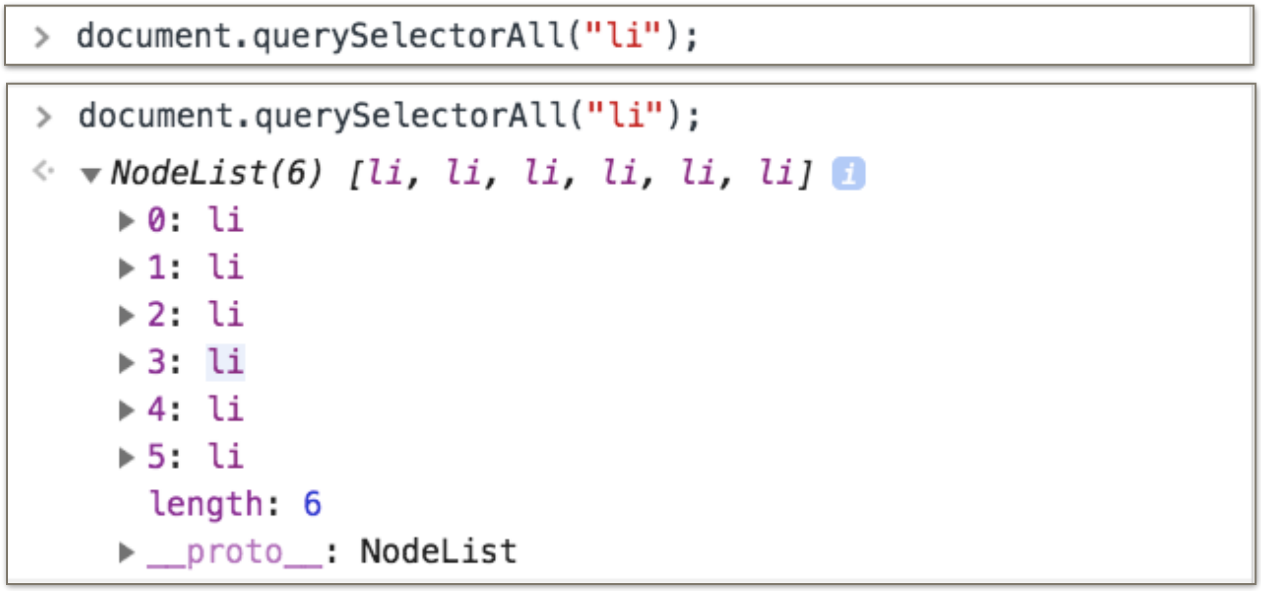
Shopping List

Get it done today

No excuses

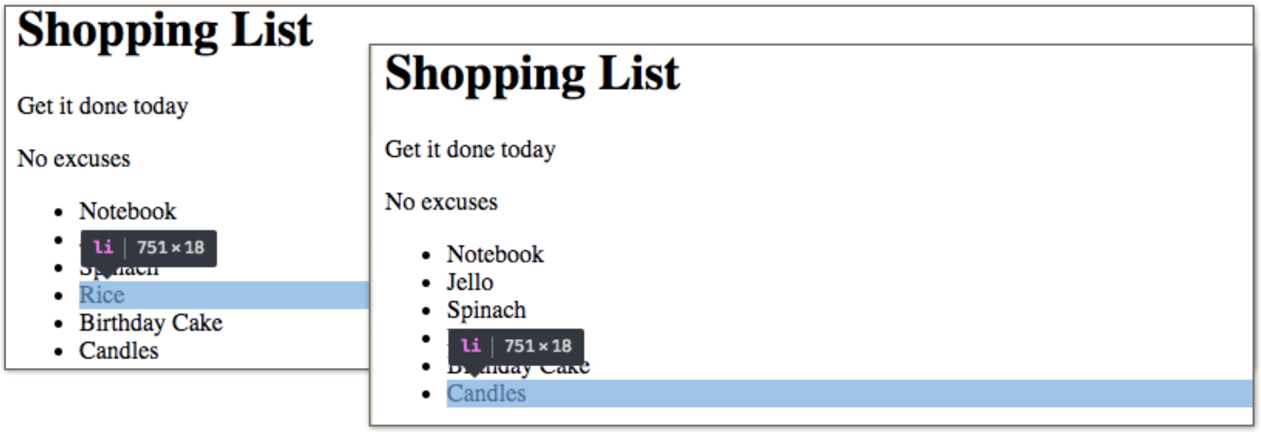
- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

querySelectorAll - selects all elements



```
> document.querySelectorAll("li");
```

```
> document.querySelectorAll("li");
<· ▼NodeList(6) [li, li, li, li, li, li] i
     ▶0: li
     ▶1: li
     ▶2: li
     ▶3: li
     ▶4: li
     ▶5: li
      length: 6
     ▶__proto__: NodeList
```



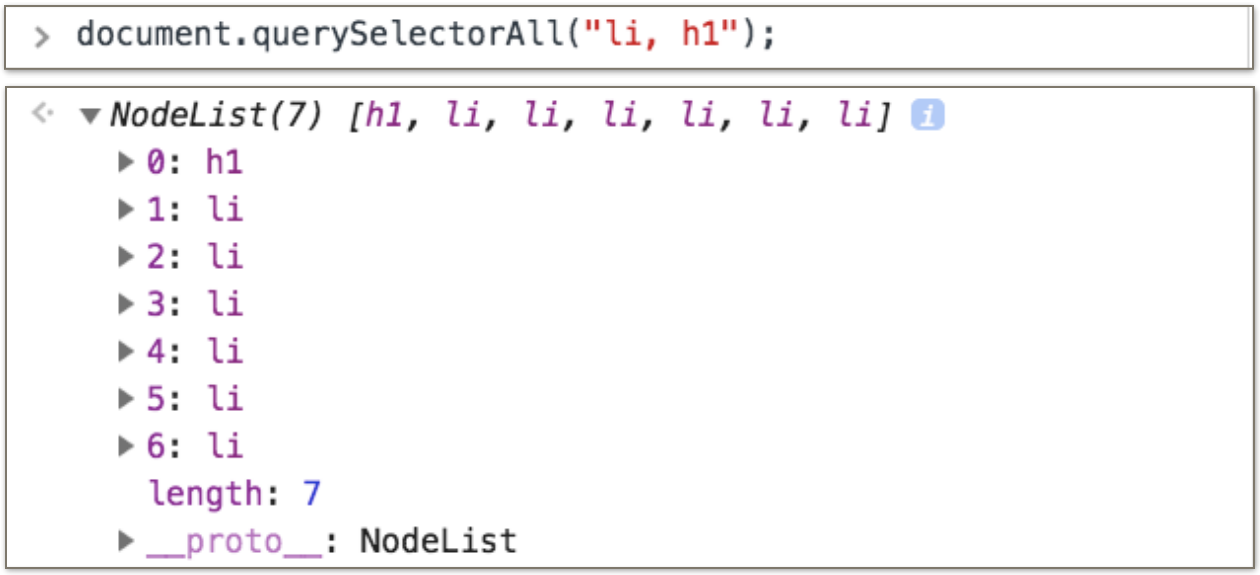
Shopping List

Get it done today

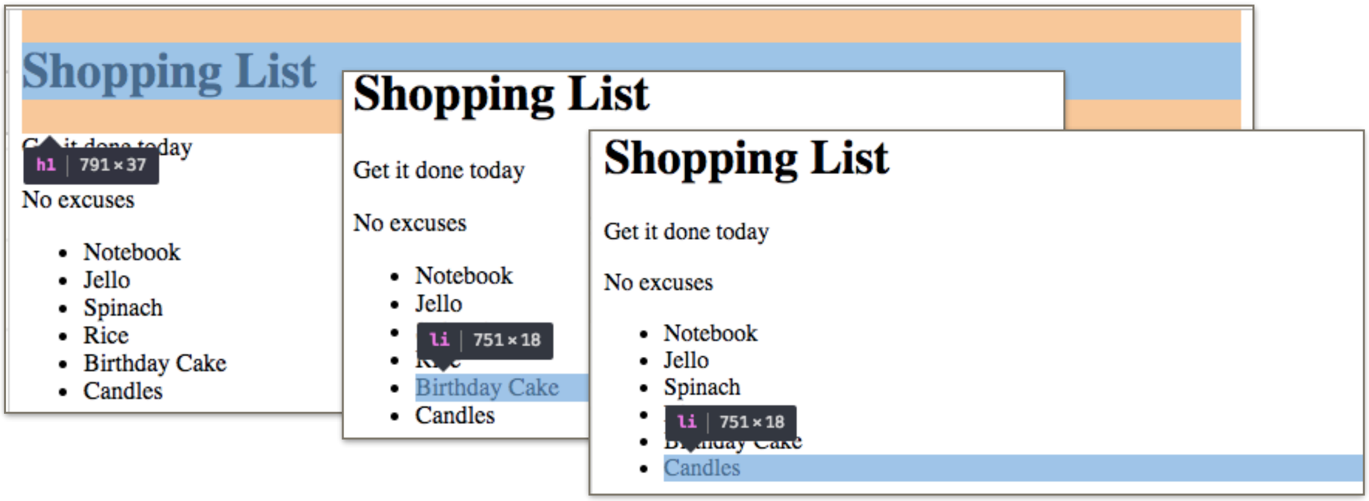
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

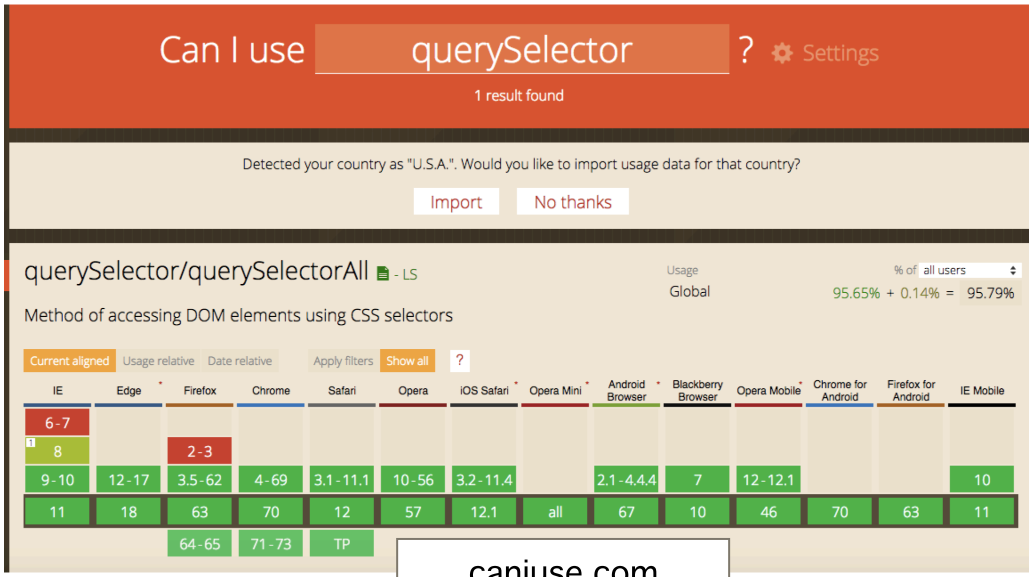
Select multiple elements, querySelectorAll



```
> document.querySelectorAll("li, h1");
<· ▼NodeList(7) [h1, li, li, li, li, li, li] i
    ▶0: h1
    ▶1: li
    ▶2: li
    ▶3: li
    ▶4: li
    ▶5: li
    ▶6: li
     length: 7
    ▶__proto__: NodeList
```



querySelector, querySelectorAll are browser compatible!



Can I use querySelector ? Settings

1 result found

Detected your country as "U.S.A.". Would you like to import usage data for that country?

Import | No thanks

querySelector/querySelectorAll - LS

Method of accessing DOM elements using CSS selectors

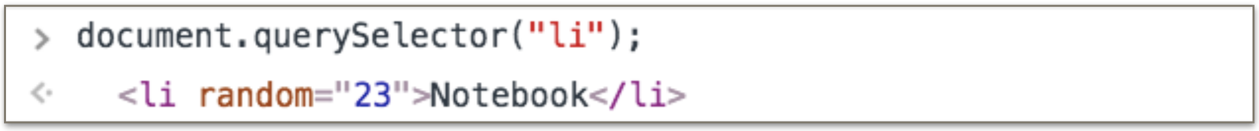
Usage: Global — % of all users: 95.65% + 0.14% = 95.79%

Current aligned | Usage relative | Date relative | Apply filters | Show all | ?

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser	Opera Mobile	Chrome for Android	Firefox for Android	IE Mobile
6-7													
8 [1]		2-3											
9-10	12-17	3.5-62	4-69	3.1-11.1	10-56	3.2-11.4		2.1-4.4.4	7	12-12.1			10
11	18	63	70	12	57	12.1	all	67	10	46	70	63	11
		64-65	71-73	TP									

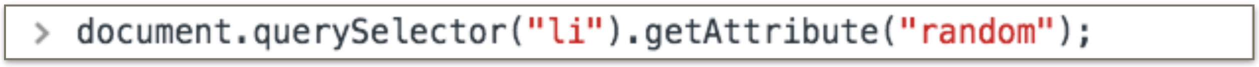
caniuse.com

Looking at a particular attribute…

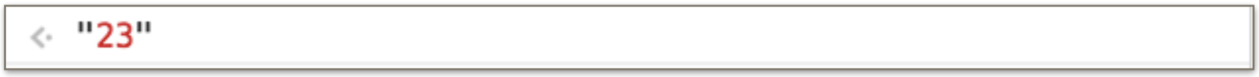


```
> document.querySelector("li");
<·   <li random="23">Notebook</li>
```

getAttribute - returns value of specified attribute

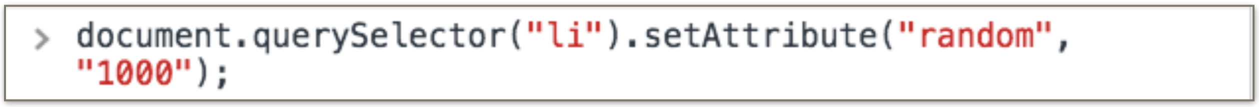


```
> document.querySelector("li").getAttribute("random");
```

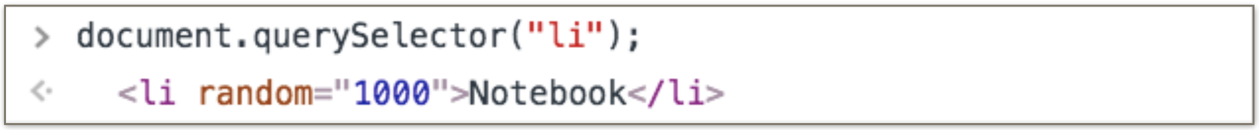


```
<· "23"
```

setAttribute - sets value of specified attribute



```
> document.querySelector("li").setAttribute("random",
  "1000");
```



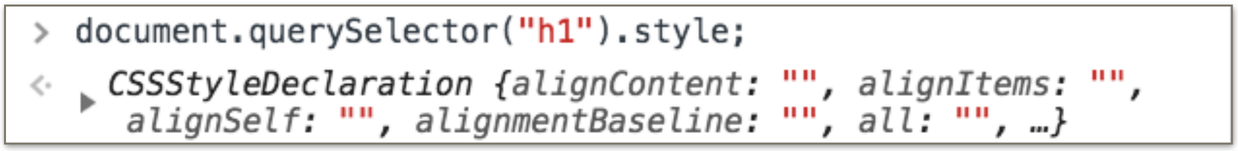
```
> document.querySelector("li");
<·   <li random="1000">Notebook</li>
```

Separation of Concerns

HTML focuses on the text
CSS focuses on the style
Javascript focuses on the actions.

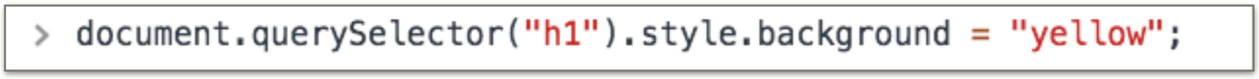
# Changing Styles # #

To see styling…

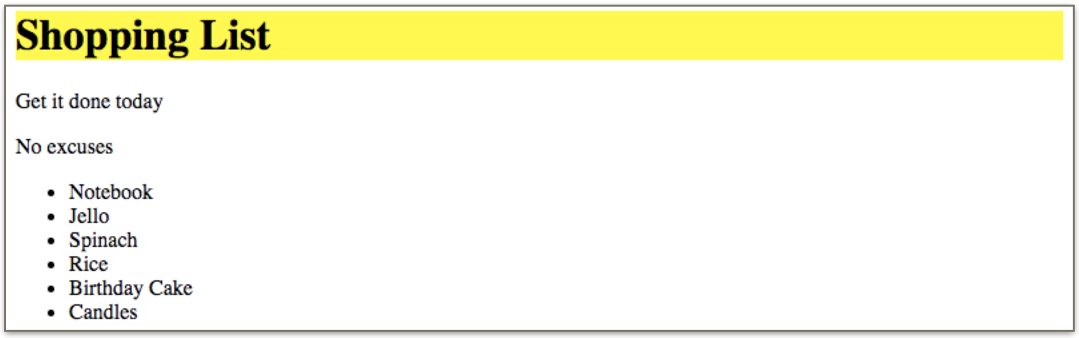


```
> document.querySelector("h1").style;
<· ▸ CSSStyleDeclaration {alignContent: "", alignItems: "",
     alignSelf: "", alignmentBaseline: "", all: "", …}
```

style . {property} - get or set in-line styling (not recommended)



```
> document.querySelector("h1").style.background = "yellow";
```



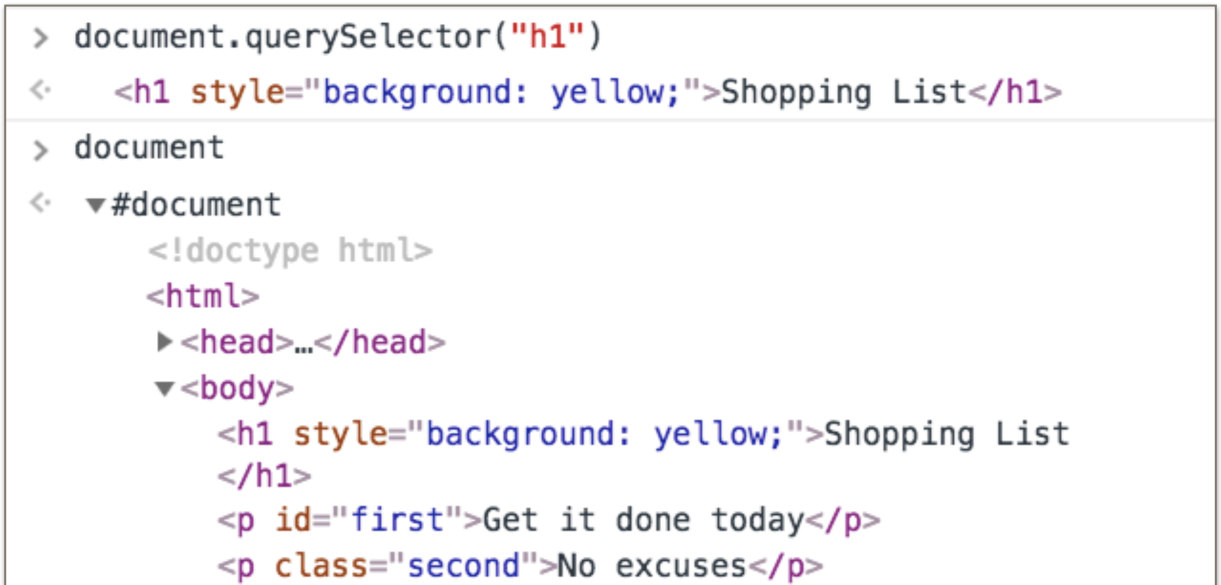
Shopping List

Get it done today

No excuses

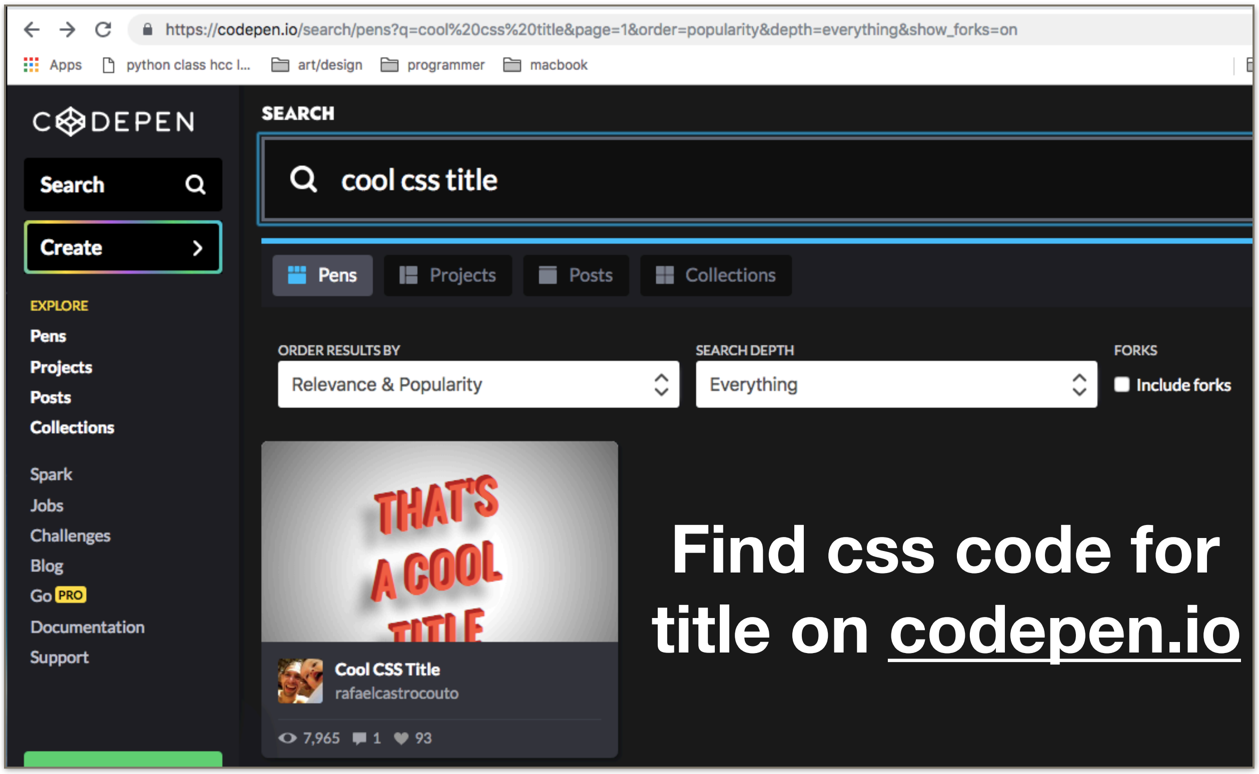
- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

messes up separation of concerns by inserting styling into HTML



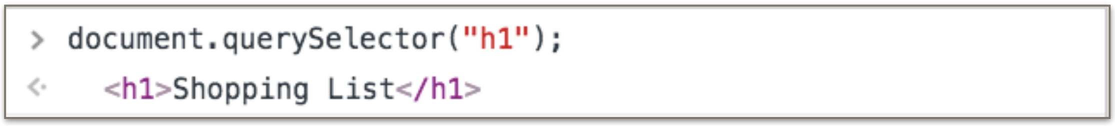
```
> document.querySelector("h1")
<·   <h1 style="background: yellow;">Shopping List</h1>
> document
<·  ▾#document
        <!doctype html>
        <html>
         ▸<head>…</head>
         ▾<body>
            <h1 style="background: yellow;">Shopping List
            </h1>
            <p id="first">Get it done today</p>
            <p class="second">No excuses</p>
```

What’s a better way?



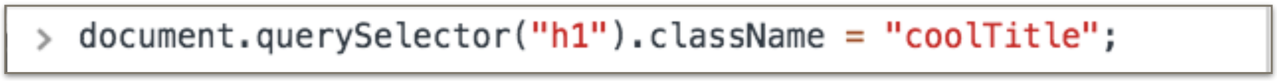
Find css code for title on codepen.io

```
             style.css

.coolTitle {
  text-align: center;
  font-family: 'Oswald', Helvetica, sans-serif;
  font-size: 40px;
  transform: skewY(-10deg);
  letter-spacing: 4px;
  word-spacing: -8px;
  color: tomato;
  text-shadow:
    -1px -1px 0 firebrick,
    -2px -2px 0 firebrick,
    -3px -3px 0 firebrick,
    -4px -4px 0 firebrick,
    -5px -5px 0 firebrick,
    -6px -6px 0 firebrick,
    -7px -7px 0 firebrick,
    -8px -8px 0 firebrick,
    -30px 20px 40px dimgrey;
}

.done {
  text-decoration: line-through;
```

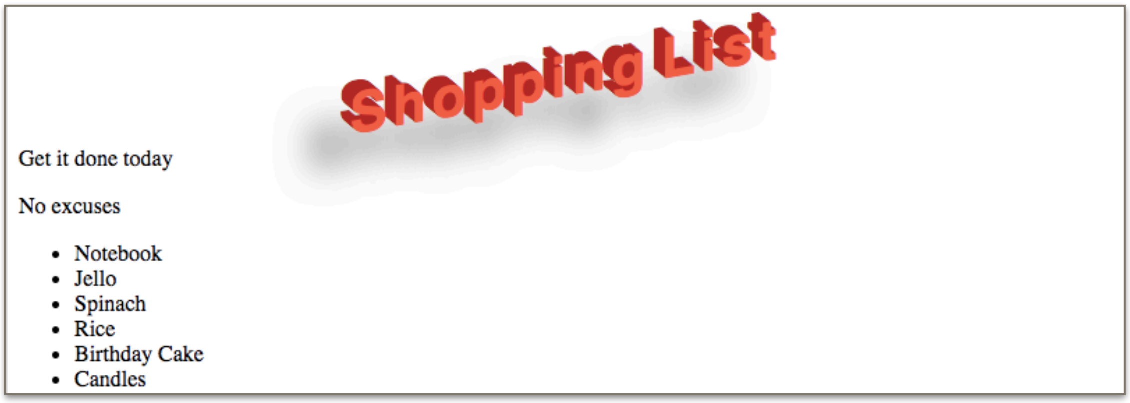


```
> document.querySelector("h1");
<·   <h1>Shopping List</h1>
```

className - sets or returns the class name of an element (from css style sheet)



```
> document.querySelector("h1").className = "coolTitle";
```

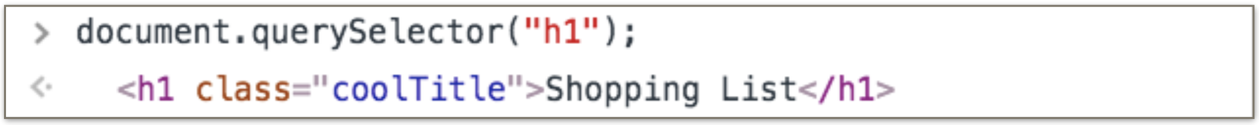


Shopping List

Get it done today

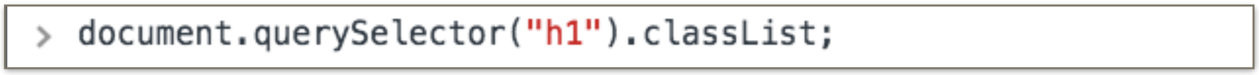
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

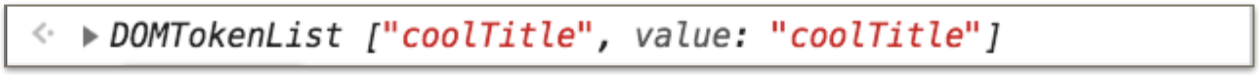


```
> document.querySelector("h1");
<·   <h1 class="coolTitle">Shopping List</h1>
```

classList - returns the class name(s) of an element

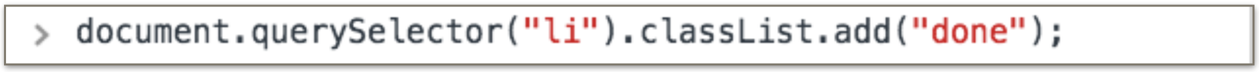


```
> document.querySelector("h1").classList;
```

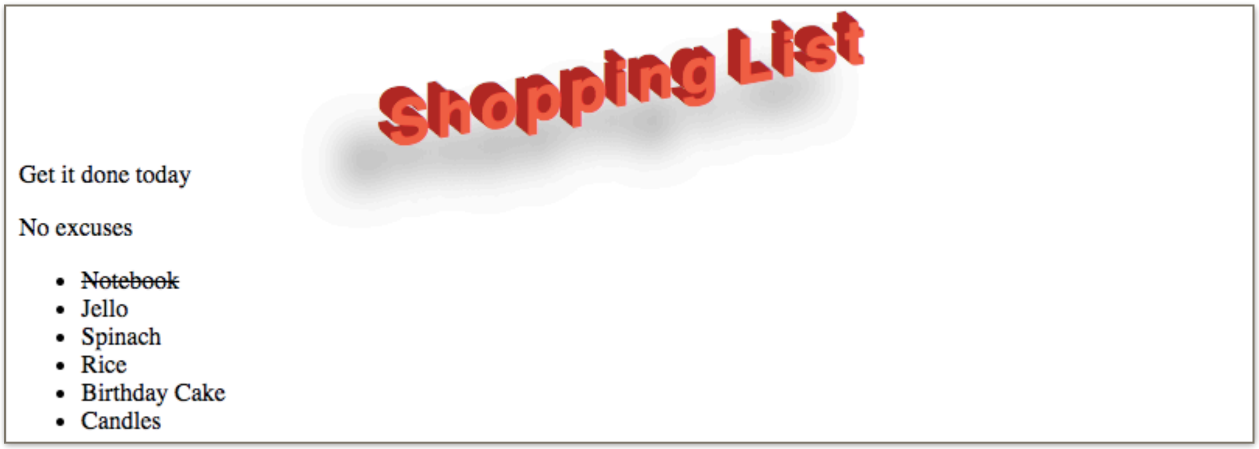


```
<· ▶ DOMTokenList ["coolTitle", value: "coolTitle"]
```

classList.add - add class name to element



```
> document.querySelector("li").classList.add("done");
```

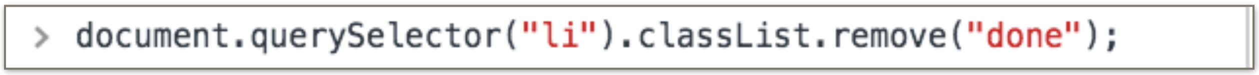


Get it done today

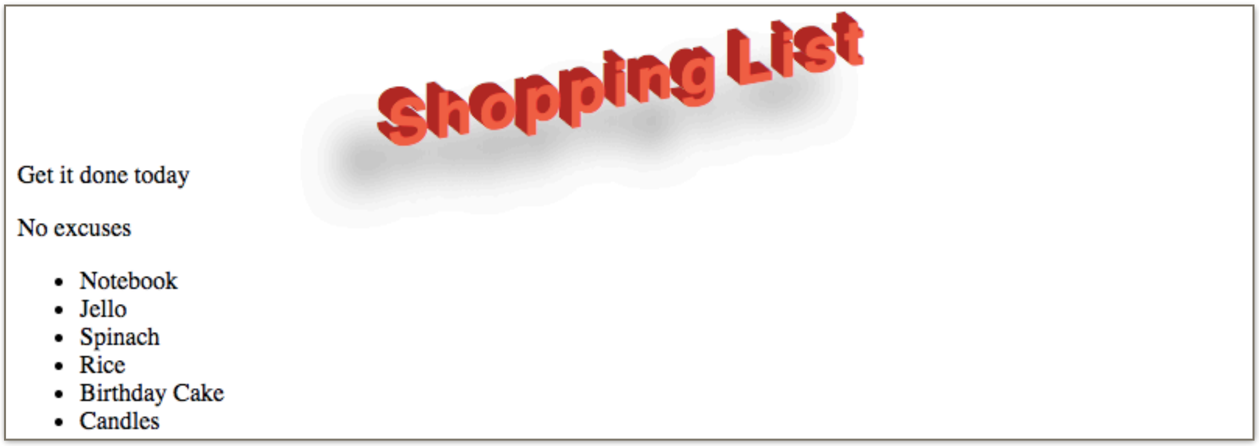
No excuses

- ~~Notebook~~
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

classList.remove - remove class name from element



```
> document.querySelector("li").classList.remove("done");
```

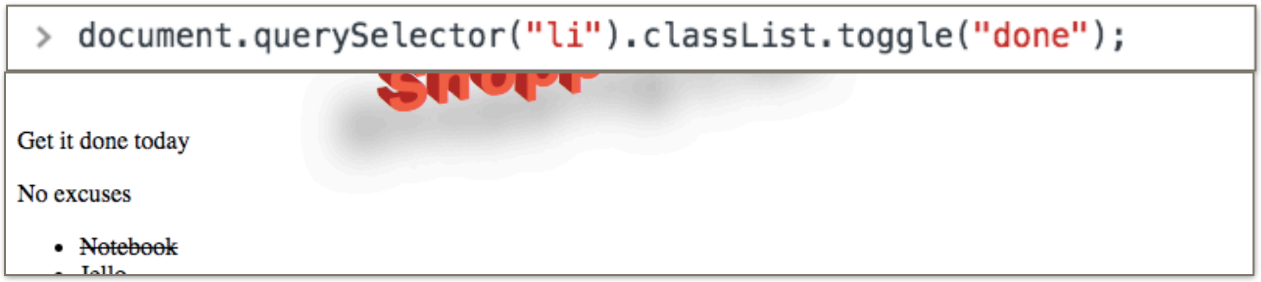


Get it done today

No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

classList.toggle - toggle class name on/off

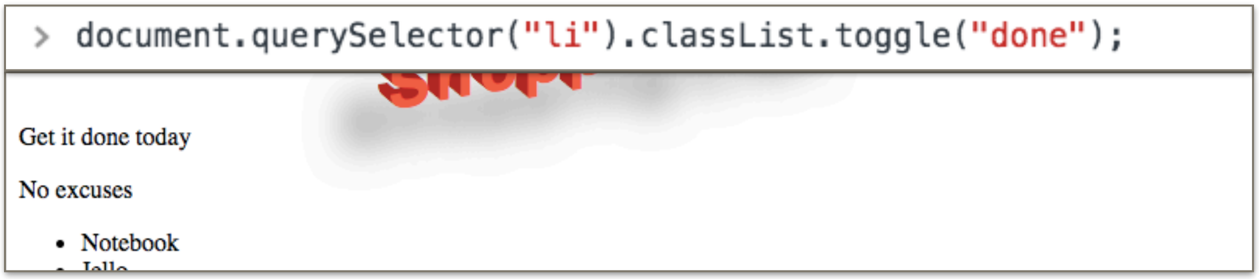


```
> document.querySelector("li").classList.toggle("done");
```

Get it done today

No excuses

- ~~Notebook~~
- Jello

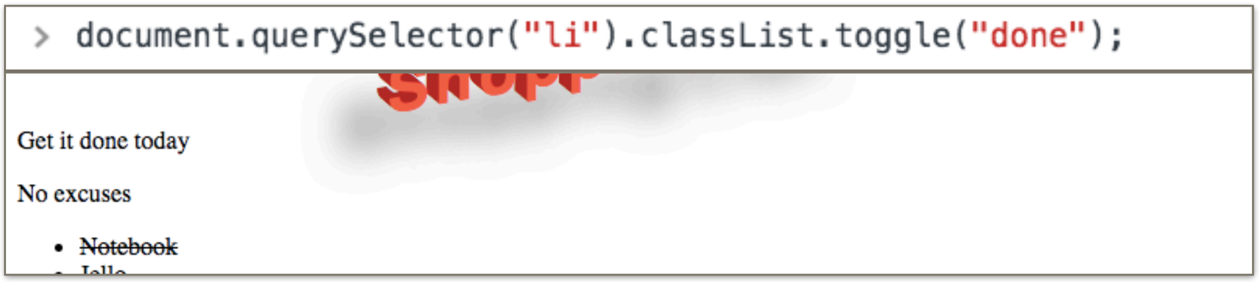


```
> document.querySelector("li").classList.toggle("done");
```

Get it done today

No excuses

- Notebook
- Jello



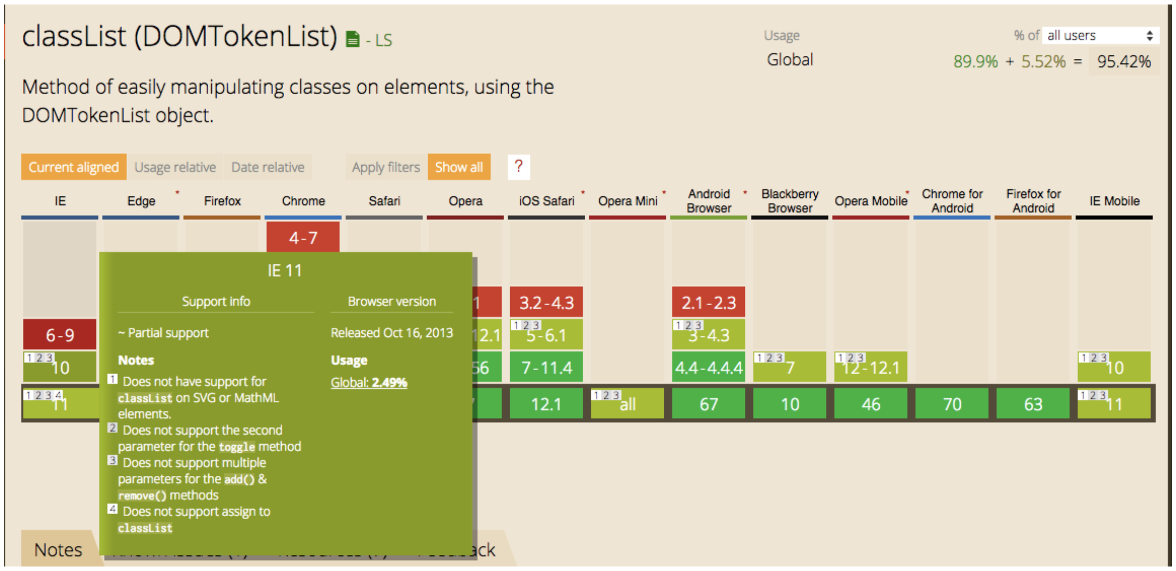
```
> document.querySelector("li").classList.toggle("done");
```

Get it done today

No excuses

- ~~Notebook~~
- Jello

Check classList on caniuse.com: partial support in ie11 so be careful. Full support for className. Get used to checking support for whatever you use in html, css, javascript!



classList (DOMTokenList) - LS

Usage Global   % of all users   89.9% + 5.52% = 95.42%

Method of easily manipulating classes on elements, using the DOMTokenList object.

IE	Edge	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	Blackberry Browser	Opera Mobile	Chrome for Android	Firefox for Android	IE Mobile
			4-7										
					1	3.2-4.3		2.1-2.3					
6-9					2.1	5-6.1[1,2,3]		3-4.3[1,2,3]					
10[1,2,3]					56	7-11.4		4.4-4.4.4	7[1,2,3]	12-12.1[1,2,3]			10[1,2,3]
11[1,2,3,4]						12.1	all[1,2,3]	67	10	46	70	63	11[1,2,3]

IE 11

Support info	Browser version
~ Partial support	Released Oct 16, 2013

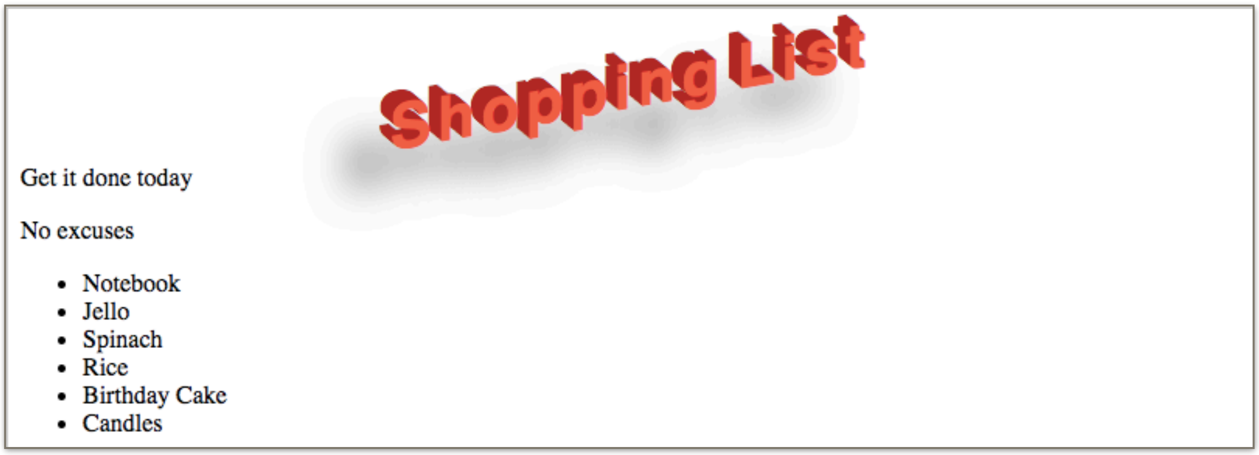
Notes

1. Does not have support for `classList` on SVG or MathML elements.
2. Does not support the second parameter for the `toggle` method
3. Does not support multiple parameters for the `add()` & `remove()` methods
4. Does not support assign to `classList`

Usage

Global: **2.49%**

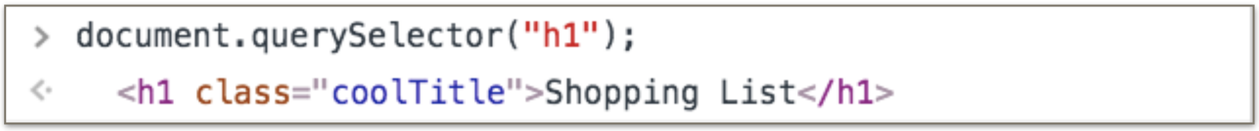
Notes



Get it done today

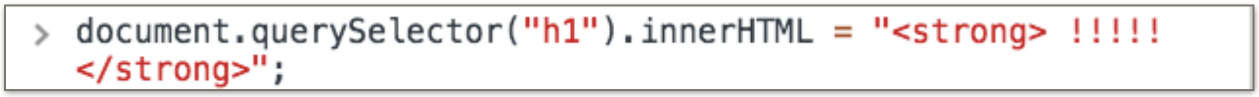
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

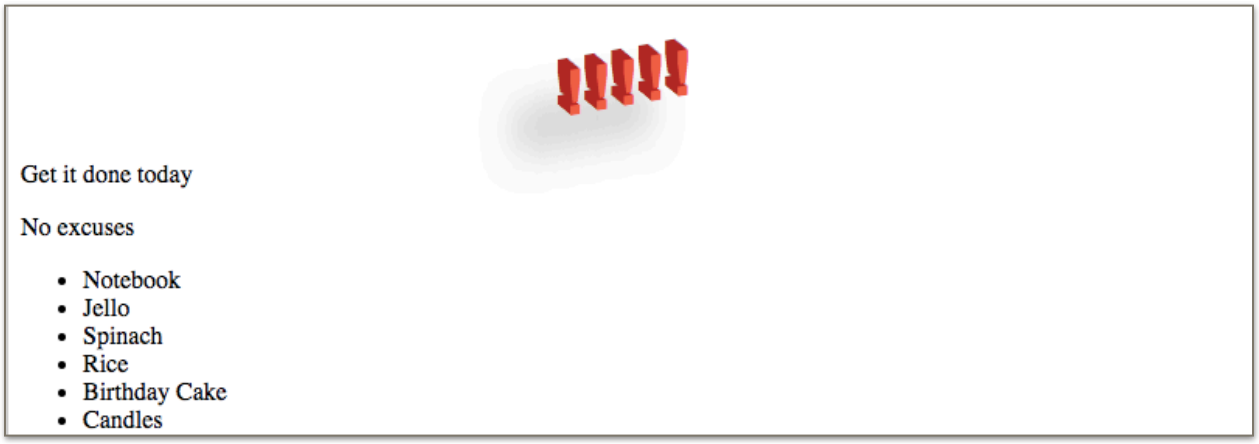


```
> document.querySelector("h1");
<·   <h1 class="coolTitle">Shopping List</h1>
```

innerHTML - sets or returns the HTML content of an element
Buggy and security issue! Do not use!!



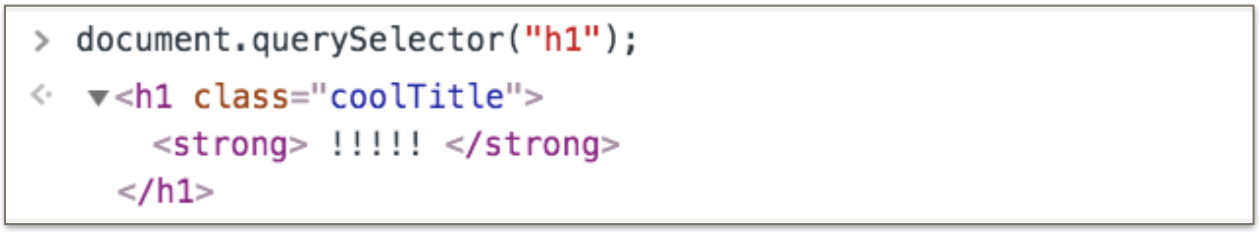
```
> document.querySelector("h1").innerHTML = "<strong> !!!!!
  </strong>";
```



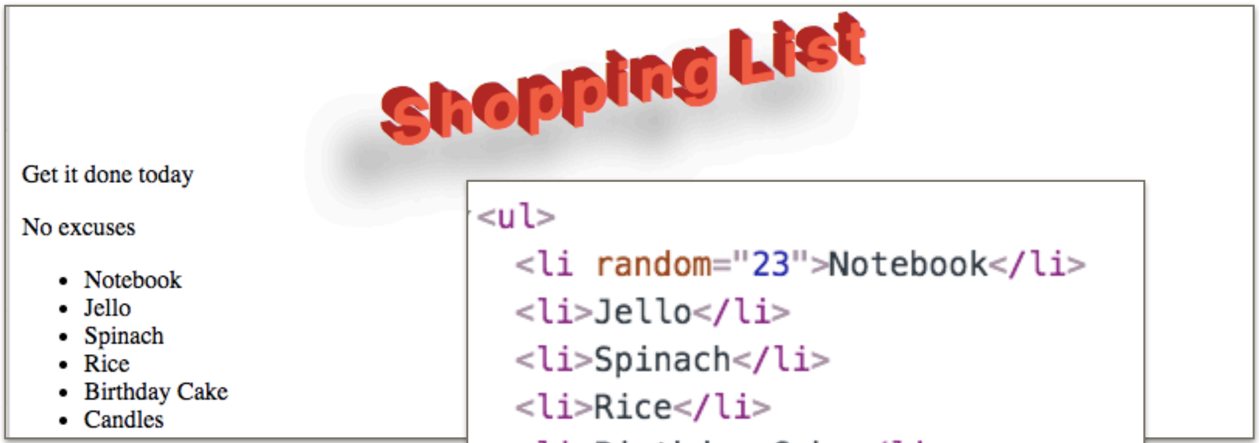
Get it done today

No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles



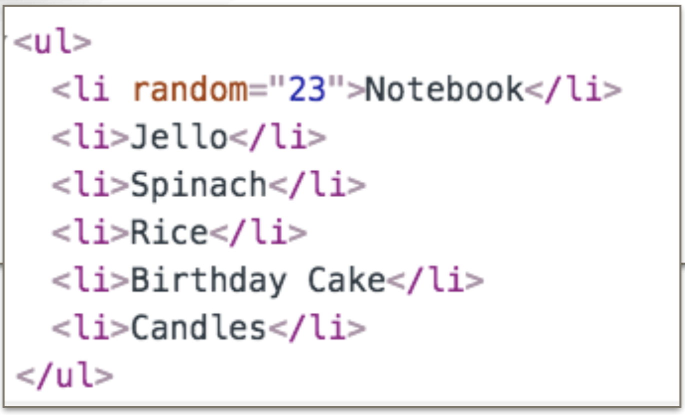
```
> document.querySelector("h1");
<· ▼<h1 class="coolTitle">
      <strong> !!!!! </strong>
    </h1>
```



Get it done today

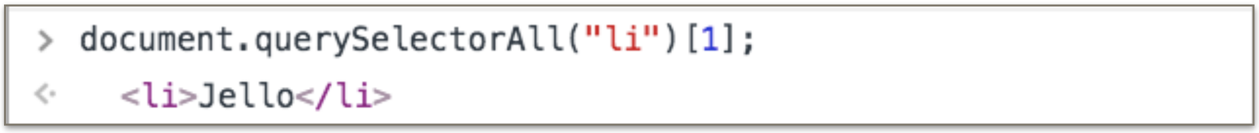
No excuses

- Notebook
- Jello
- Spinach
- Rice
- Birthday Cake
- Candles

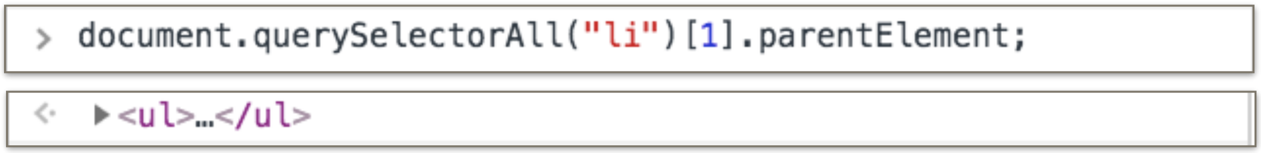


```
<ul>
  <li random="23">Notebook</li>
  <li>Jello</li>
  <li>Spinach</li>
  <li>Rice</li>
  <li>Birthday Cake</li>
  <li>Candles</li>
</ul>
```

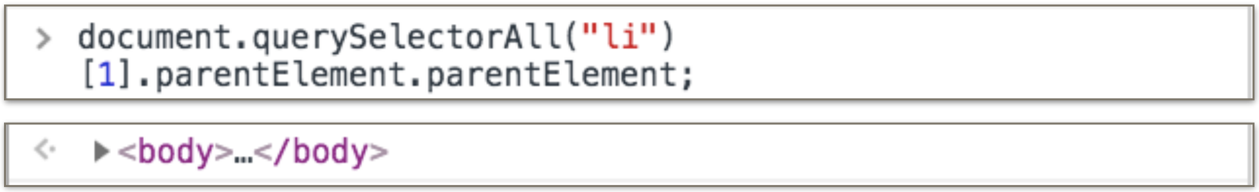
parentElement - returns parent element of specified element



```
> document.querySelectorAll("li")[1];
<·   <li>Jello</li>
```

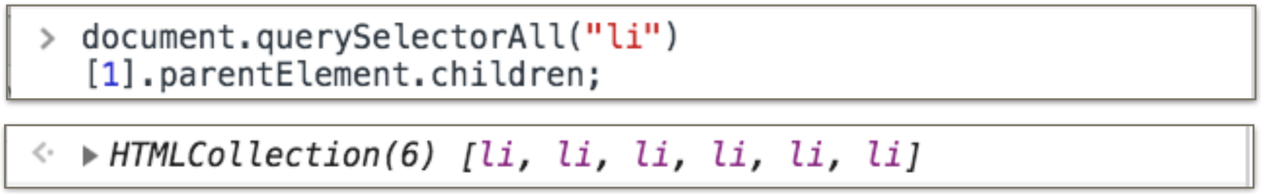


```
> document.querySelectorAll("li")[1].parentElement;
<·  ▶<ul>…</ul>
```



```
> document.querySelectorAll("li")
  [1].parentElement.parentElement;
<·  ▶<body>…</body>
```

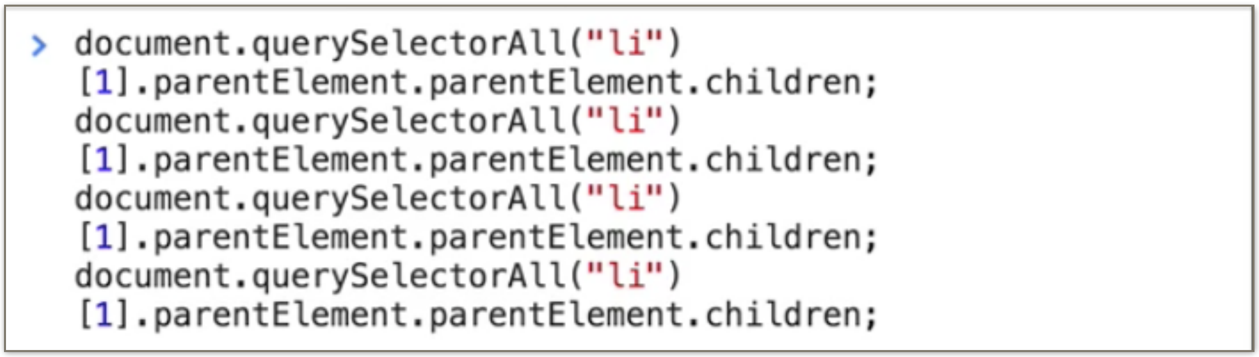
children - returns a collection of an element's child elements



```
> document.querySelectorAll("li")
  [1].parentElement.children;
<· ▶HTMLCollection(6) [li, li, li, li, li, li]
```

Cache Selectors - store in variable (redundant actions)

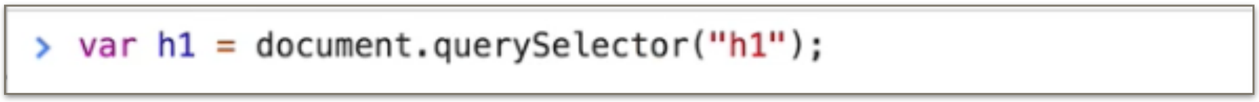
Let’s say we have a javascript file and we're selecting things whenever we want to use them.



```
> document.querySelectorAll("li")
  [1].parentElement.parentElement.children;
  document.querySelectorAll("li")
  [1].parentElement.parentElement.children;
  document.querySelectorAll("li")
  [1].parentElement.parentElement.children;
  document.querySelectorAll("li")
  [1].parentElement.parentElement.children;
```

The web browser has to look up the DOM, find it, and store it in memory each time we select it. The Web browser is doing actions over and over, using memory each time, even though it is selecting the same thing each time.

Instead, let’s do this:



```
> var h1 = document.querySelector("h1");
```

So now any time I need to use H1, the Web browser doesn't have to go look up the DOM, find H1, and store it in memory.

Now we have H1 living in the variable until we refresh the page, so that Web browser’s work is done.