4. Introducing jQuery Filters and Selectors

Client-Side Web Programming

José Socuéllamos

Index

1.- Introducing jQuery

"There are only two kind of languages: the ones people complain about and the ones nobody uses"

Bjarne Stroustrup – C++ Designer

- Some JS libraries out there: Ajax, Prototype, Node.js, jQuery, etc…

1.- Introducing jQuery

- jQuery is used by 73% of the Top Million websites.
- It was created in 2006 to simplify the client-side scripting.

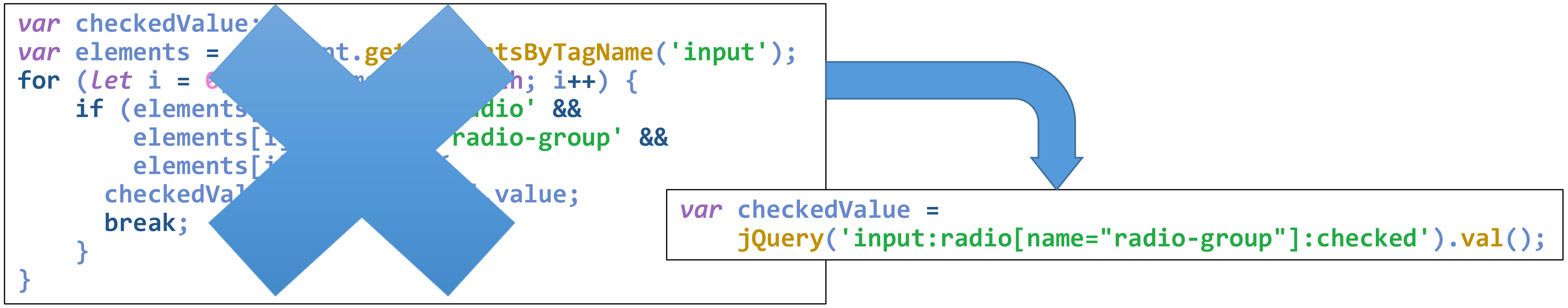
“WRITE LESS, DO MORE”

```
var checkedValue;
var elements = document.getElementsByTagName('input');
for (let i = 0; i < elements.length; i++) {
    if (elements[i].type === 'radio' &&
        elements[i].name === 'radio-group' &&
        elements[i].checked) {
      checkedValue = elements[i].value;
      break;
    }
}
```

1.- Introducing jQuery

- jQuery is used by 73% of the Top Million websites.
- It was created in 2006 to simplify the client-side scripting.

"WRITE LESS, DO MORE"



```
var checkedValue;
var elements = [illegible]nt.get[illegible]tsByTagName('input');
for (let i = 0[illegible]h; i++) {
    if (elements[illegible]dio' &&
        elements[i[illegible]radio-group' &&
        elements[i[illegible]
      checkedVa[illegible].value;
      break;
    }
}
```

```
var checkedValue =
    jQuery('input:radio[name="radio-group"]:checked').val();
```

https://api.jquery.com/

2.- Downloading jQuery

- Download jQuery from the official website (jQuery.com/download).
  - Uncompressed file: best used during development or debugging.
  - ★ Compressed file (min): saves bandwidth and improves performance in production.
- Save it in your local machine and link it from your webpage.

```
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>jQuery Hello World DAW</title>
    <script type="text/javascript" src="js/jquery-3.7.1.min.js"></script>
</head>
<body>

</body>
</html>
```

2.- Linking jQuery

- Include jQuery from a CDN (Content Delivery Network).
  - For example, Google, Microsoft…

```html
<!DOCTYPE html>
<head>
    <meta charset="UTF-8">
    <title>jQuery Hello World DAW</title>
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
    <script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-3.7.1.min.js"></script>
</head>
<body>

</body>
</html>
```

3.- Hello World

- In JavaScript we use the `window.onload` event to wrap all of our code.

```
window.onload = function () {

    alert("The web page is loaded!!!")

}
```

- The `onload` event is triggered when all the content of the page has been loaded (including images).

3.- Hello World

- In jQuery we will write all of our code inside a `document.ready` event.

```
$(document).ready(function () {

    alert("The web page is loaded!!!");

});
```

```
$(function () {

    alert("The web page is loaded!!!");

});
```

- This will prevent any jQuery code from running before the document is finished loading (is ready).
- It will also allow us to have our JavaScript code before the body of our document, in the head section.

3.- Hello World

- If our jQuery code is in an external file, we can also load it asynchronously by adding the `defer` attribute in the *script* tag.
- The script will load in the background and run once the DOM is complete.
- We can use this attribute to load jQuery too.

```
<script src="jquery/3.7.1/jquery.min.js" defer></script>
<script src="resources/js/mycode.js" defer></script>
```

4.- Selecting Elements

- Selectors allows us to get content from the document and manipulate it.
- They are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes….
- They return a jQuery object with multiple functions and properties to interact with.
- We have several selectors in jQuery:
  - Simples
  - Composite
  - Filters

http://api.jquery.com/category/selectors/

4.- Selecting Elements

- Basic selectors are based on the CSS syntax and work basically the same way.

Selector	Description	Example
tag	Gets the elements with the specified HTML tag	$("div");
#id	Gets the element with the specified id	$("#myimg");
.class	Gets the elements with the specified class name	$(".myclass");
tag.class	Gets the elements with the specified HTML tag and class	$("ul.customclass");
tag#id.class	Gets the elements with the specified HTML tag, id and class	$("form#input1.myclass");
*	Gets all elements in the page	$("*");

- We can check the selection with .length

```
if ($('div.foo').length) { ... }
```

4.- Selecting Elements

- Composite selectors allow to get objects by its hierarchy and combination.
- Some of them are:

Selector	Description	Example
E, F, G…	Gets all the specified elements	$("p, ul.b");
E>F…	Gets all F elements that are direct children of E	$("ul.customclass>li>a");
E F	Gets all F elements that are descendants of E	$("table td")
E+F	Gets all F elements that are immediately preceded by sibling E	$("p+div")
E~F	Gets all F elements preceded by any sibling E	$("p~div")
.class1.class2	Gets the elements with class1 and class2	$(".a.b");

5.- Basic Filters

- Filters keep the simplicity of selecting elements in jQuery and are used to polish the results of a selector.
- There are many types of filters, but these are some of the basic ones:

Filter	Description	Example
:first	Gets the first element	$("div:first"); $("ul li:first"); //first ul first li
:last	Gets the last element	$("div:last");
:even // :odd	Gets the even/odd elements	$("div:even"); $("div:odd");
:eq(n)//:gt(n)//:lt(n)	Gets the elements equal, greater or lower than the specified index (starts at 0)	$("div:eq(3)"); $("div:gt(6)"); $("div:lt(4)");
:not(selector)	All the elements but the ones that meet the provided selector	$("div:not(div:eq(2))");

6.- Advanced Filters

- Attribute filters
  - They allow us to refine the results gathered by the selector using the attributes of the element.
  - Attribute selectors are extremely powerful and allow you to select elements based on their attributes.
  - You can easily recognize these selectors because they're wrapped with square brackets (for example, [selector]).
  - They can be very slow.

6.- Advanced Filters

- Attribute filters
  - We can have multiple filters working as an AND. [filter][filter]

Filter	Description	Example
[attributeName]	Get element that contain a specified attribute	$("form[method]");
[attributeName=value]	Get the element with the given attribute and with the given value. You can also use !=.	$("div[id='container']");
[attributeName^=value]	Get the element with the given attribute and with the value beginning with the given value. You can also use !^	$("div[id^='container']");
[attributeName$=value]	Get the element with the given attribute and with the value finishing with the given value. You can also use !$	$("a[href$='.pdf']");
[attributeName*=value]	Get the element with the given attribute and with the value containing the given value. You can also use !*	$("a[href*='jquery.com']");

6.- Advanced Filters

- Content filters
  - They allow us to refine the results gathered by the selector using the content of the element.

Filter	Description	Example
:contains(text)	Gets elements that contains the specified text	$("div:contains('my house')");
:empty	Gets all elements that are empty.	$("div:empty");
:has(selector)	Gets all elements that contain the specified selector.	$("div:has(p[class=a])");
:parent	Gets all elements that are a parent of another element (containing at least one element)	$("div:parent");

6.- Advanced Filters

- Type filters
  - They allow us to refine the results gathered by the selector using the type of the element.

Filter	Description	Example
:header	All header elements <h1>, <h2> ...	$(":header")
:animated	All elements that are in the progress of an animation	$(":animated")

6.- Advanced Filters

- Visibility filters
  - They allow us to refine the results gathered by the selector depending if the elements are visible or not.

Filter	Description	Example
:visible	Get the visible elements	$("div:visible");
:hidden	Get the hidden elements	$("div:hidden");

  - What's a hidden element?
    - Set to display:none
    - Form elements with type="hidden"
    - Width and height set to 0
    - A hidden parent element (this also hides child elements)
    - Note: It will not work on elements with visibility:hidden.

6.- Advanced Filters

- Child filters
  - They allow us to refine the results gathered by the selector considering its relationship with their parents.

Filter	Description	Example
:nth-child(index)	The element at the specified index	$("div p:nth-child(2)");

:nth-child(even)
:nth-child(odd)
	Even/odd elements	$("div p:nth-child(even)");
		$("div p:nth-child(odd)");

:first-child :last-child	Get first/last child of a element	$("div p:first-child"); $("div p:last-child");
:only-child	Get the child without siblings	$("div p:only-child");

6.- Advanced Filters

- Form filters
  - Very similar to the previous filters, but useful to find specific elements in a form.

Filter	Description	Example
:button	Gets all button elements and input elements with type="button"	$(":button");
:checkbox	Gets all input elements with type="checkbox"	$(":checkbox")
:file	Gets all input elements with type="file"	$(":file")
:image	Gets all input elements with type="image"	$(":image")
:input	Gets all form elements (input, select, textarea, button)	$(":input")
:password	Gets all input elements with type="password"	$(":password")
:radio	Gets all input elements with type="radio"	$(":radio")
:reset // :submit	Gets all elements with type="reset" // type="submit" (buttons and inputs)	$(":reset") $(":submit")
:text	Gets all input elements with type="text" or without a type specified (type="text" is the default)	$(":text")

$("input[type='button']");

.
.
.

6.- Advanced Filters

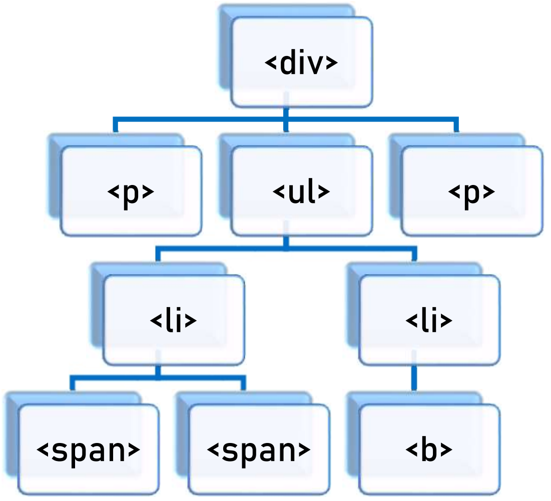
- Form filters
  - Very similar to the previous filters, but useful to find specific elements in a form.

Filter	Description	Example
:checked	Gets all checked input elements (checkboxes, radio and options of select)	$(":checked")
:disabled	Gets all disabled input elements	$(":disabled")
:enabled	Gets all enabled input elements	$(":enabled")
:focus	Gets the element that has the focus at the time the selector is run	$(":focus")
:selected	Gets all selected options in a select element	$(":selected")

- Very similar to the previous filters, but useful to find specific elements in a form.

7.- jQuery Traversing

- jQuery traversing are used to select HTML elements based on their relation to other elements.
- We can start with one selection and move through that selection until we reach the elements we desire.
- In the right image, we can easily move up (ancestors), down (descendants) and sideways (siblings) in the tree, starting from the selected (current) element.
  - <div> is the **parent** of <ul>, and an **ancestor** of everything inside of it
  - <ul> is the **parent** of both <li>, and a **child** of <div>
  - left <li> is the **parent** of <span>, **child** of <ul> and a **descendant** of <div>
  - <span> is a **child** of the left <li> and a **descendant** of <ul> and <div>
  - both <li> are **siblings** (they share the same **parent**)
  - right <li> is the **parent** of <b>, **child** of <ul> and a **descendant** of <div>
  - <b> is a **child** of right <li> and a **descendant** of <ul> and <div>



7.- jQuery Traversing

Filter	Description	Example
.children()	Gets the children of each element (optionally filtered by a selector)	$("ul").children()
.closest()	For each element in the set, gets the first element that matches the selector (going up)	$("span").closest("ul")
.find()	Gets the descendants of each element, filtered by a selector, jQuery object, or element	$("div").find("li")
.next()	Gets the immediately following sibling of each element (optional selector)	$("p:first").next()
.nextAll()	Gets all following siblings of each element in the set (optionally filtered by a selector)	$("p:first").nextAll()
.nextUntil()	Get all following siblings of each element up to the selected element	$("p:first").nextUntil("p")
.parent()	Gets the parent of each element in the current set (optionally filtered by a selector)	$("b").parent()
.parents()	Gets the ancestors of each element (optionally filtered by a selector)	$("b").parents()
.prev()	Gets the immediately preceding sibling of each element (optional selector)	$("p:last").prev()
.prevAll()	Gets all preceding siblings of each element (optionally filtered by a selector)	$("p:last").prevAll()
.prevUntil()	Get all preceding siblings of each element up to the selected element	$("p:last").prevUntil("p")
.siblings()	Gets the siblings of each element in the set (optionally filtered by a selector)	$("ul").siblings()

8.- Using JS variables in jQuery selectors

- We can use JavaScript variables as a parameter in a jQuery selector.

```
var par = prompt("Enter the paragraph number");
console.log($("pe:eq(" + par + ")"));
```

- Using the "+" symbol we can concatenate the selector with the variable, inserting it in the right place.

```
var cla = prompt("Enter the class name");
console.log($("." + cla));
```