

SET

A JavaScript Set is a collection of unique values.

```
const letters = new Set();
letters.add("a");
letters.add("b");
*****

const letters = new Set(["a","b","c"]);
document.getElementById("demo").innerHTML = letters.size;
*****

const letters = new Set();
letters.add("a");
letters.add("b");
letters.add("c");
letters.add("c");
letters.add("c");
letters.add("c");
letters.add("c");
letters.add("c");
letters.add("c");

document.getElementById("demo").innerHTML = letters.size;  => 3
*****

const letters = new Set(["a","b","c"]);
let text = "";
```

```
letters.forEach (function(value) {  
  text += value + "<br>";  
})  
*****  
const letters = new Set(["a","b","c"]);  
let text = "";  
for (const x of letters.values()) {  
  text += x + "<br>";  
}  
*****
```

MAP

A Map holds key-value pairs where the keys can be any datatype.

```
const fruits = new Map([  
  ["apples", 500],  
  ["bananas", 300],  
  ["oranges", 200]  
]);  
  
document.getElementById("demo").innerHTML = fruits.get("apples");  
*****
```

```
const fruits = new Map([
  ["apples", 500],
  ["bananas", 300],
  ["oranges", 200]
]);
fruits.set("apples", 200);
document.getElementById("demo").innerHTML = fruits.get("apples");
*****

const fruits = new Map([
  ["apples", 500],
  ["bananas", 300],
]);
document.getElementById("demo").innerHTML = fruits.get("apples");
*****
```

```
const fruits = new Map([
```

```
["apples", 500],  
["bananas", 300],  
]);  
fruits.delete("apples");  
document.getElementById("demo").innerHTML = fruits.has("apples");  
*****
```

typeof

```
typeof "John"           // Returns "string"  
typeof 3.14              // Returns "number"  
typeof NaN               // Returns "number"  
typeof false             // Returns "boolean"  
typeof [1,2,3,4]         // Returns "object"  
typeof {name:'John', age:34} // Returns "object"  
typeof new Date()        // Returns "object"  
typeof function () {}    // Returns "function"  
typeof myCar              // Returns "undefined" *  
typeof null              // Returns "object"
```

JQUERY

The purpose of jQuery is to make it much easier to use JavaScript on your website.

Cdn:

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
```

Styntax:

```
$(document).ready(function(){  
$(selector).action()  
});
```

```
*****
```

```
$("*")
```

```
$(this)
```

```
$("p.intro")
```

```
$("#test")
```

```
....
```

Methods : css , attr, addClass, removeClass, toggleClass, text, html, val

```
*****
```

Graphic methods => hide, fade, slide, fadeTo

Practice : menu Responsive, popUp , webpack-sess40/index6-textSize.html

SELECTORS

<u>*</u>	\$("#*")	All elements
<u>#id</u>	\$("#lastname")	The element with id="lastname"
<u>.class</u>	\$(".intro")	All elements with class="intro"
<u>.class.class</u>	\$(".intro,.demo")	All elements with the class "intro" or "demo"

<u>element</u>	<code>\$("p")</code>	All <p> elements
<u>el1,el2,el3</u>	<code>\$("h1,div,p")</code>	All <h1>, <div> and <p> elements
<u>:first</u>	<code>\$("p:first")</code>	The first <p> element
<u>:last</u>	<code>\$("p:last")</code>	The last <p> element
<u>:even</u>	<code>\$("tr:even")</code>	All even <tr> elements
<u>:odd</u>	<code>\$("tr:odd")</code>	All odd <tr> elements

:first-child	<code>\$("p:first-child")</code>	All <p> elements that are the first child of their parent
:first-of-type	<code>\$("p:first-of-type")</code>	All <p> elements that are the first <p> element of their parent
:last-child	<code>\$("p:last-child")</code>	All <p> elements that are the last child of their parent
:last-of-type	<code>\$("p:last-of-type")</code>	All <p> elements that are the last <p> element of their parent
:nth-child(<i>n</i>)	<code>\$("p:nth-child(2)")</code>	All <p> elements that are the 2nd child of their parent
:nth-last-child(<i>n</i>)	<code>\$("p:nth-last-child(2)")</code>	All <p> elements that are the 2nd child of their parent, counting from the last child
:nth-of-type(<i>n</i>)	<code>\$("p:nth-of-type(2)")</code>	All <p> elements that are the 2nd <p> element of their parent

[:nth-last-of-type\(*n*\)](#)

\$("p:nth-last-of-type(2)")

All <p> elements that are the 2nd <p> element of their parent, counting from the last child

[:only-child](#)

\$("p:only-child")

All <p> elements that are the only child of their parent

[:only-of-type](#)

\$("p:only-of-type")

All <p> elements that are the only child, of its type, of their parent

[parent > child](#)

\$("div > p")

All <p> elements that are a direct child of a <div> element

[parent descendant](#)

\$("div p")

All <p> elements that are descendants of a <div> element

[element + next](#)

\$("div + p")

The <p> element that are next to each <div> elements

<u>element ~ siblings</u>	<code>\$("div ~ p")</code>	All <p> elements that appear after the <div> element
---	----------------------------	--

<u>:eq(<i>index</i>)</u>	<code> \$("ul li:eq(3)")</code>	The fourth element in a list (index starts at 0)
--	---------------------------------	--

<u>:gt(<i>no</i>)</u>	<code> \$("ul li:gt(3)")</code>	List elements with an index greater than 3
---------------------------------------	---------------------------------	--

<u>:lt(<i>no</i>)</u>	<code> \$("ul li:lt(3)")</code>	List elements with an index less than 3
---------------------------------------	---------------------------------	---

<u>:not(<i>selector</i>)</u>	<code> \$("input:not(:empty)")</code>	All input elements that are not empty
--	---------------------------------------	---------------------------------------

<u>:header</u>	<code> \$(" :header")</code>	All header elements <h1>, <h2> ...
--------------------------------	------------------------------	------------------------------------

:animated	<code>\$(":animated")</code>	All animated elements
:focus	<code>\$(":focus")</code>	The element that currently has focus
:contains(<i>text</i>)	<code>\$(":contains('Hello')")</code>	All elements which contains the text "Hello"
:has(<i>selector</i>)	<code>\$("div:has(p)")</code>	All <div> elements that have a <p> element
:empty	<code>\$(":empty")</code>	All elements that are empty
:parent	<code>\$(":parent")</code>	All elements that are a parent of another element
:hidden	<code>\$("p:hidden")</code>	All hidden <p> elements
:visible	<code>\$("table:visible")</code>	All visible tables

:root	<code>\$(":root")</code>	The document's root element
:lang(<i>language</i>)	<code>\$("p:lang(de)")</code>	All <p> elements with a lang attribute value starting with "de"
[<i>attribute</i>]	<code>\$("[href]")</code>	All elements with a href attribute
[<i>attribute</i>= <i>value</i>]	<code>\$("[href='default.htm']")</code>	All elements with a href attribute value equal to "default.htm"
[<i>attribute</i>!= <i>value</i>]	<code>\$("[href!='default.htm']")</code>	All elements with a href attribute value not equal to "default.htm"
[<i>attribute</i>\$= <i>value</i>]	<code>\$("[href\$='.jpg']")</code>	All elements with a href attribute value ending with ".jpg"

[attribute=value]	<code>\$("[title='Tomorrow']")</code>	All elements with a title attribute value equal to 'Tomorrow', or starting with 'Tomorrow' followed by a hyphen
[attribute^=value]	<code>\$("[title^='Tom']")</code>	All elements with a title attribute value starting with "Tom"
[attribute~=value]	<code>\$("[title~='hello']")</code>	All elements with a title attribute value containing the specific word "hello"
[attribute*=value]	<code>\$("[title*='hello']")</code>	All elements with a title attribute value containing the word "hello"
:input	<code>\$(":input")</code>	All input elements
:text	<code>\$(":text")</code>	All input elements with type="text"

:password	<code>\$(":password")</code>	All input elements with type="password"
:radio	<code>\$(":radio")</code>	All input elements with type="radio"
:checkbox	<code>\$(":checkbox")</code>	All input elements with type="checkbox"
:submit	<code>\$(":submit")</code>	All input elements with type="submit"
:reset	<code>\$(":reset")</code>	All input elements with type="reset"
:button	<code>\$(":button")</code>	All input elements with type="button"
:image	<code>\$(":image")</code>	All input elements with type="image"
:file	<code>\$(":file")</code>	All input elements with type="file"

:enabled	<code>\$(":enabled")</code>	All enabled input elements
:disabled	<code>\$(":disabled")</code>	All disabled input elements
:selected	<code>\$(":selected")</code>	All selected input elements
:checked	<code>\$(":checked")</code>	All checked input elements

EVENT

- dblclick
- mouseenter

- mouseleave
- mousedown
- mouseup
- hover
- focus
- focusin
- focusout
- blur
- on
- change
- `$(document).mousemove(function(event){`
- `$("#span").text("X: " + event.pageX + ", Y: " + event.pageY);`
- `});`
- `$("#a").click(function(event){`
- `event.preventDefault();`
- `});`
- keydown
- keypress
- keyup
 - `function para(event){`
 - `document.getElementsByTagName('h1')[0].innerHTML=event.which || event.keyCode`
 - `}`
- one
- scroll
- scrollTop
- scrollLeft()
- select
- submit

JQUERY ANIMATE AND STOP

CHAINING AND CALLBACK

append() - Inserts content at the end of the selected elements

prepend() - Inserts content at the beginning of the selected elements

after() - Inserts content after the selected elements

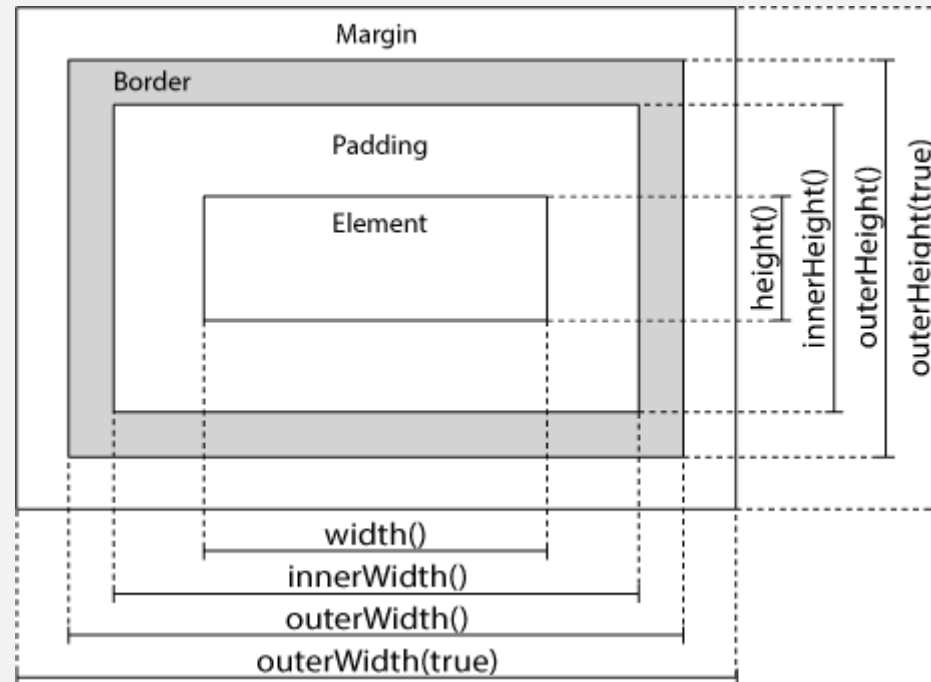
before() - Inserts content before the selected elements

practice : webpack-sess40/index4-mina.html

methods => remove , empty

Dimensions

- width()
- height()
- innerWidth()
- innerHeight()
- outerWidth()
- outerHeight()



Traversing

- `add()`
- `children()`
- `each()`
- `eq()`
- `filter()`
- `find()`
- `has()`
- `next()`
- `nextAll()`
- `nextUntil()`

- prev()
- prevAll()
- prevUntil()
- parent()
- parents()
- parentsUntil()
- offsetParent()
- siblings()
- not()
- slice()
- first()
- last()

noConflict()

ANOTHER

- clone()
- detach()
- insertAfter()

- insertBefore()
- offset()
- position()
- prop()
- removeProp()
- replaceAll()
- unwrap()
- wrap()
- wrapAll()
- wrapInner()
- length

practice : webpack-sess41/index9-keyword.html