


WUM



¿Que dice el
libro que es el
DOM?

DOM es una abreviatura
de **Document Object Model**

Podríamos traducirlo por
***Modelo de Objeto de
Documento.***

O podemos referirnos al DOM
con el nombre de ***jerarquía de
objetos del navegador***

¿Qué quiere decir?



Es la forma en la que esta estructurado un documento XML.

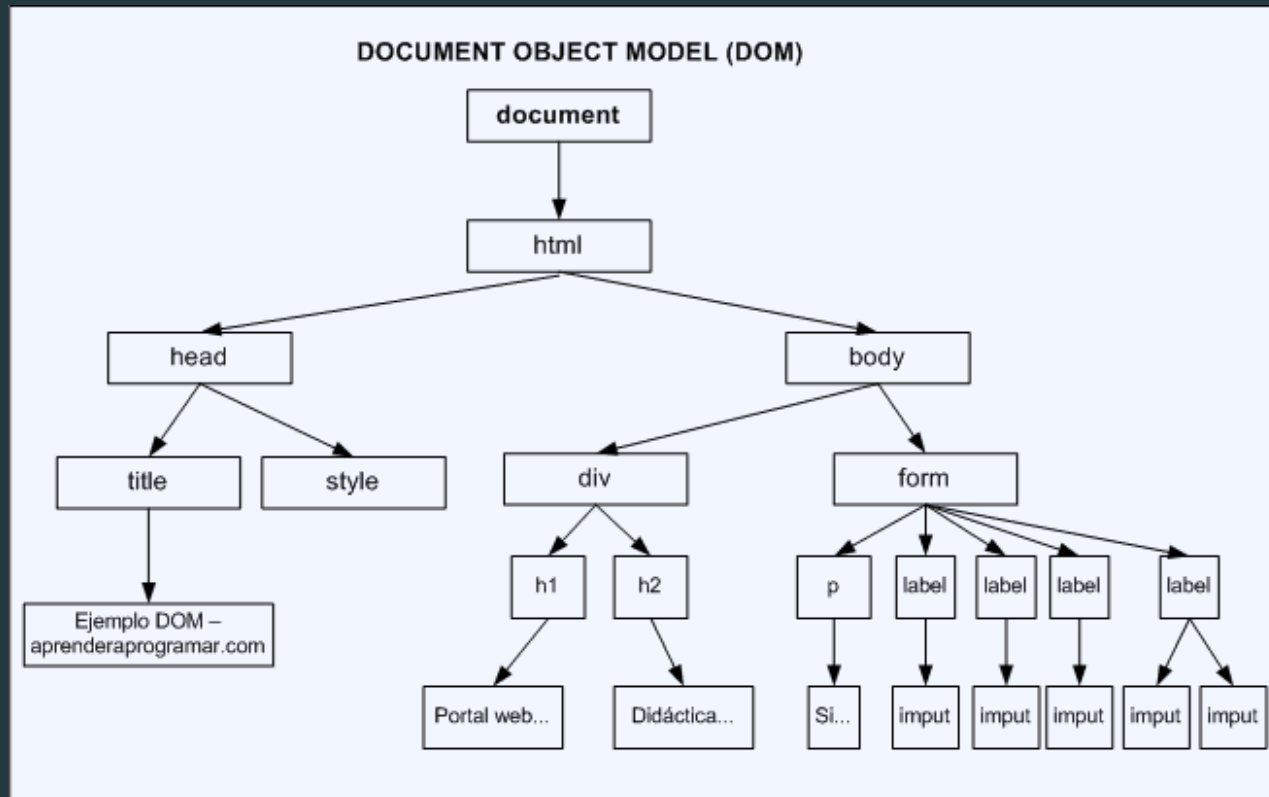


Un documento HTML es un documento XML



Es la forma en la que esta estructurado un documento HTML

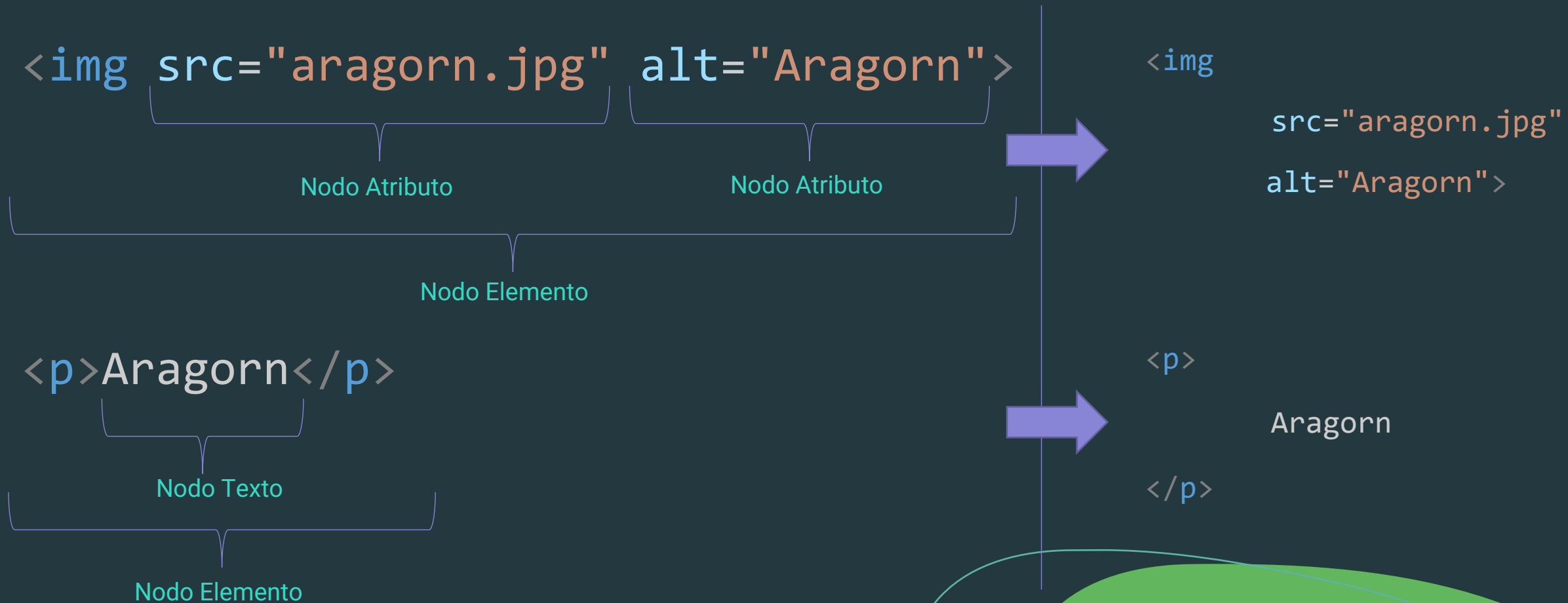
¿Qué es el árbol del DOM?



En el modelo DOM para XML, se considera un nodo a **cualquier cosa** que se encuentra dentro del documento:

- Todo el documento en su conjunto es un **nodo documento**.
- Cada elemento XML es un **nodo elemento**.
- El texto de los elementos XML son **nodos texto**.
- Cada atributo es un **nodo atributo**.
- Los comentarios son **nodos comentario**.

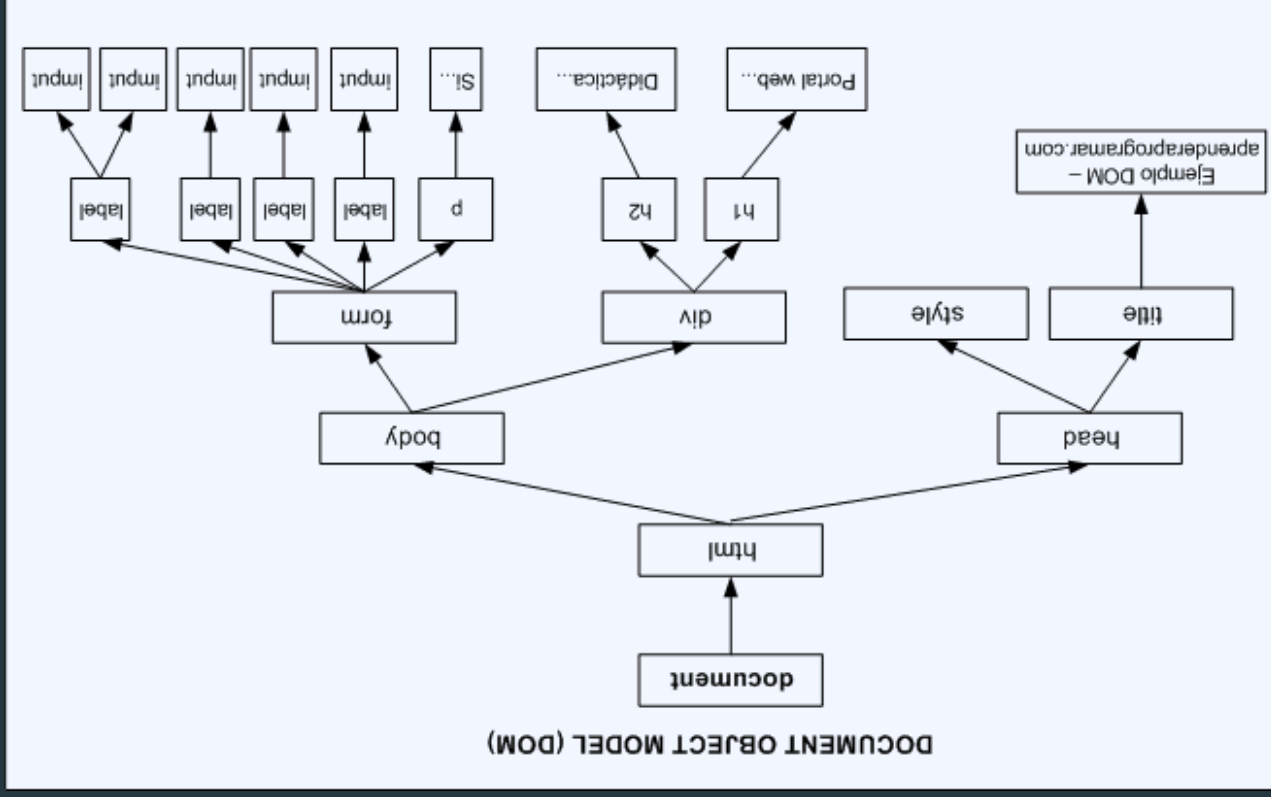
Un nodo es...

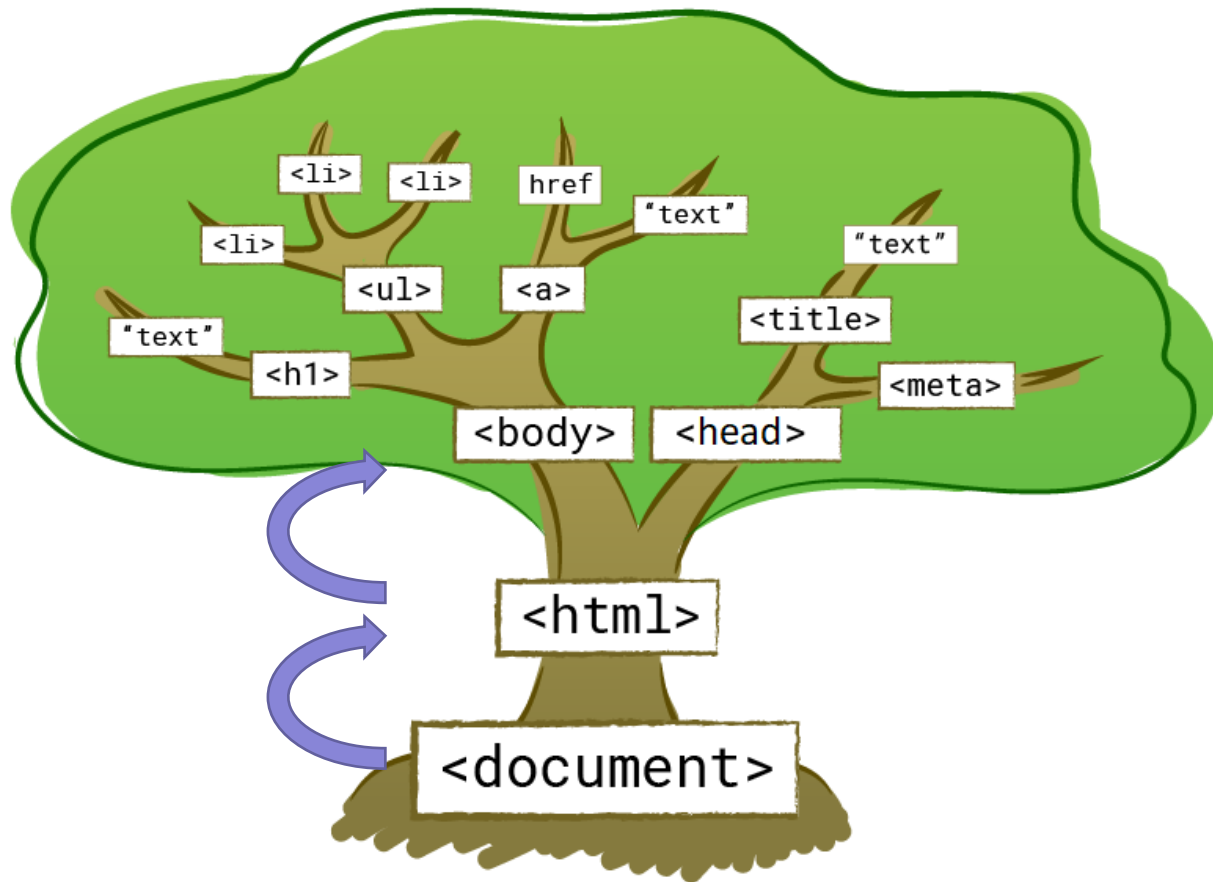




El modelo DOM surgió como una especificación para permitir que los programas Java y los scripts de JavaScript fueran portables entre los navegadores web.

<https://www.tokioschool.com/noticias/que-es-modelo-dom-modelo-objetos-documento/>

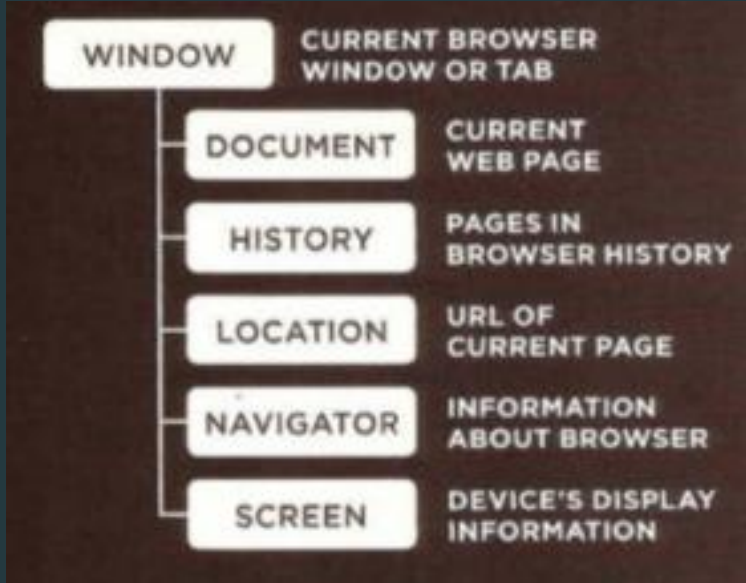




- document es el padre de html
- html es el padre de body y de head
- document es el padre DEL PADRE de body
- document es el padre del padre del padre de h1

Window: El padre de todos

Propiedades



window.innerHeight - altura (excluding browser chrome/user interface) (px)

window.innerWidth - anchura (excluding browser chrome/user interface) (px)

window.pageXOffset - Distance document has been scrolled horizontally (px)

window.pageYOffset - Distance document has been scrolled vertically (px)

window.screenX - Coordenada X, respecto esquina superior-izquierda (px)

window.screenY - Coordenada Y, respecto esquina superior-izquierda (px)

window.location - URL actual (o ruta local al archivo)

window.document - Referencia al objeto documento, usado para representar la pagina actual contenida en windows

window.history - Referencia al objeto history de la pestaña o ventana del navegador que contiene detalles de las paginas que han sido vistas en esa ventana o pestaña

window.history.length - numero de historias en el objeto history para esa pestaña o ventana del navegador

window.screen - Hace referencia al objeto screen del monitor

window.screen.width - Anchura de la pantalla del monitor(px)

window.screen.height - Altura de la pantalla del monitor (px)

<https://jolav.me/chuletas/javascript-para-web/>

Acceso al DOM: selectores básicos

Selector	Sintaxis	Ejemplo	Significado
Universal	*	* { margin-left:0; }	En todos los elementos de la página, margen izquierdo a cero.
Etiqueta	Etiqueta	p { color:grey; }	En todos los párrafos (p), fuente de color gris.
De Clase	.clase	.menu { margin: 0 auto; }	En los elementos de la clase " menu", centrados respecto a izda y dcha.
De ID	#id	#destacado { font-size: 12px; }	En los elementos con ID "destacado", tamaño de fuente 12 pixels.
Descendente	selector selector	p a { text-decoration:none; }	En los enlaces que se encuentran dentro de párrafos, no aparece el subrayado.

Acceso al DOM: selectores avanzados

Selector	Sintaxis	Ejemplo	Significado
Hijo directo	Padre > hijo	Div > p { color: blue; }	El párrafo SOLO tendrá la letra azul si es hijo directo (no nieto)
Selector adyacente	Elemento + elemento	p + img { border:2px solid orange; }	Todas las img que vayan a continuación de otro párrafo , no dentro (hermanos, no hijos)
Selector de atributos	Elemento[atributo="valor"]	img[alt="Gimli">{ border-radius: 15px;	Todas las imagenes que tengan un atributo alt cuyo valor sea "Gimli"
	[atributo=valor]	[width = "400"]{ Display: none; }	Todo lo que tenga un atributo width cuyo valor sea "400"
	[atributo ~= valor]	[class ~= "invisible"]{ Display: none; }	Todo lo que tenga un atributo class, y uno de sus valores sea "invisible"
	[atributo = valor]	[lang = "en-"]{ Display: none; }	Todo lo que tenga un atributo Lang cuyo valor comience por "en-"

Acceso al DOM: PseudoElementos

Selector	Sintaxis	Ejemplo	Significado
antes de	::before	section::before { content: url(gimli.jpg); }	Inserta una foto antes de cada sección de la página
Después de	::after	section::after { content: url(gimli.jpg); }	Inserta una foto después de cada sección de la página
Primera letra	::first-letter	P::first-letter { Color: Yellow }	La primera letra de todos los párrafos en amarillo
Primera linea	::first-line	p + p::first-line { color: greenyellow; }	Cambia el tamaño solo a la primera letra de cada párrafo
Marcador de lista	::marker	li::marker { content: "=>" }	Cambia el bullet de la lista

Acceso al DOM: PseudoClases

Selector	Sintaxis	Ejemplo	Significado
a no visitado	a:link	a:link {color: blue;}	Los links no visitados serán de color azul
a visitado	a:visited	a:link {color: red;}	Los links visitados serán de color rojo
Pasar con el ratón	elemento:hover	a:hover {color: Orange;}	Cuando se pase el ratón por encima del link se volverá naranja
a selección	a:active	a:active{ color: green;}	Al clicar en un link se volverá verde

Acceso al DOM: PseudoClases

Selector	Sintaxis	Ejemplo	Significado
Primer hijo	elemento:first-child	article:first-child{ color: yellow; }	El article que sea primer hijo tendrá la letra amarilla.
X hijo	Elemento:nth-child(n)	Article:nth-child(2){ Font-weight: bold;}	El segundo nodo hijo de article en negrita.
Ultimo hijo	elemento:last-child	ul :last-child{ color: blue; }	El último nodo hijo de ul con letra azul.
Por tipo	Elemento:nth-of-type(n)	img:nth-of-type(3){ Width: 20%;}	La tercera imagen (de cada padre) 20% de ancho
Por descarte	:not(selector)	:not(img){ background:black; }	Todo lo que no sea una imagen, en fondo negro

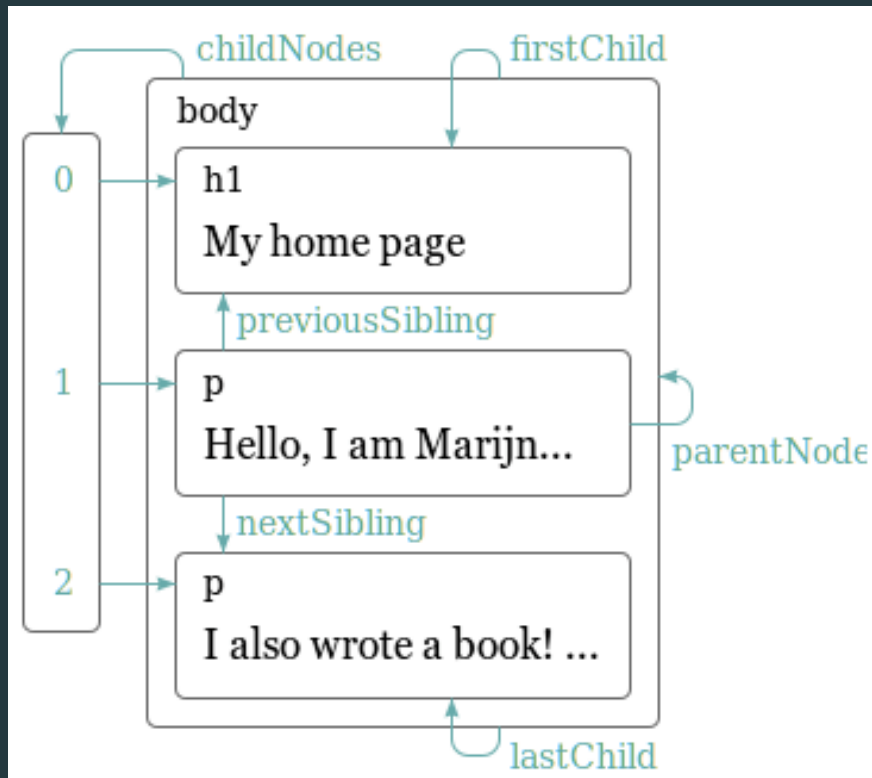
https://www.w3schools.com/css/css_pseudo_classes.asp

Manipulación del DOM

Hay 3 formas de hacerlo:

1. `document.write()`
2. `Element.innerHTML`
3. Manipulación del DOM

El objeto document



Seleccionar un elemento

document.getElementById("name") - Selecciona el elemento con id="name"

document.querySelector("h4") - Selecciona el primer elemento h4

element.parentNode - Toma el padre del elemento actual

element.previousSibling - Coge el hermano previo

element.nextSibling - Coge el hermano siguiente

element.firstChild - Coge el primer hijo del actual elemento

element.lastChild - Coge el ultimo hijo del actual elemento

Seleccionar varios elementos : Los metodos devuelven una NodeList

document.getElementsByClassName("name") - Selecciona todos los elementos que tienen la clase="name"

document.getElementsByTagName("li") - Coge todos los elementos que tienen el tag name="li"

document.querySelectorAll(h4) - Selecciona todos los elementos que poseen h4

Trabajar con elementos

element.nodeValue - Permite acceder al contenido de un nodo texto

```
document.getElementById("one").firstChild.nextSibling.nodeValue;
```

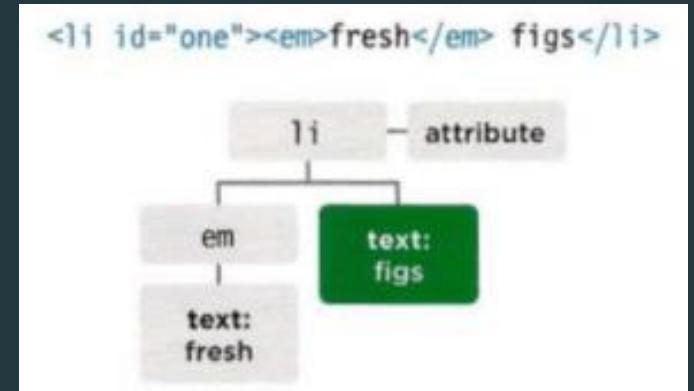
element.innerHTML - Permite acceder a elementos hijo y contenido de texto y de markup

```
var elContent = document.getElementById('one').innerHTML; // Get
// elContent = "<em>fresh</em> figs"
document.getElementById('one').innerHTML = elContent; // Set
```

element.textContent - Permite acceder al texto del elemento y de sus hijos

element.className - Valor del atributo class

element.id - Valor del atributo id



Empieza la magia...

```
var newEl = document.createElement("li");  
var newText = document.createTextNode("quinoa");  
newEl.appendChild(newText);  
// Ahora lo posicionamos donde le corresponda  
var position = document.getElementsByTagName("ul")[0];  
position.appendChild(newEl);
```

document.createElement() - Crea nodos

document.createTextNode() - Crea nodos de texto

element.appendChild() - Añade nodos

element.removeChild() - Elimina nodo

element.hasAttribute(n) - Chequea si existe el atributo n

element.getAttribute(n) - Consigue el valor del atributo n

element.setAttribute(n) - Pone el valor del atributo n

element.removeAttribute(n) - Elimina el atributo n

The most common DOM methods at a glance

Reaching Elements In a Document

`document.getElementById('id')`: Retrieves the element with the given `id` as an object

`document.getElementsByTagName('tagName')`: Retrieves all elements with the tag name `tagName` and stores them in an array-like list

Reading Element Attributes, Node Values and Other Data

`node.getAttribute('attribute')`: Retrieves the value of the attribute with the name `attribute`

`node.setAttribute('attribute', 'value')`: Sets the value of the attribute with the name `attribute` to `value`

`node.nodeType`: Reads the type of the `node` (1 = element, 3 = text node)

`node.nodeName`: Reads the name of the `node` (either element name or `#textNode`)

`node.nodeValue`: Reads or sets the value of the `node` (the text content in the case of text nodes)

Navigating Between Nodes

`node.previousSibling`: Retrieves the previous sibling node and stores it as an object.

`node.nextSibling`: Retrieves the next sibling node and stores it as an object.

`node.childNodes`: Retrieves all child nodes of the object and stores them in an list. here are shortcuts for the first and last child node, named `node.firstChild` and `node.lastChild`.

`node.parentNode`: Retrieves the node containing `node`.

Creating New Nodes

`document.createElement(element)`: Creates a new element node with the name `element`. You provide the name as a string.

`document.createTextNode(string)`: Creates a new text node with the node value of `string`.

`newNode = node.cloneNode(bool)`: Creates `newNode` as a copy (clone) of `node`. If `bool` is `true`, the clone includes clones of all the child nodes of the original.

`node.appendChild(newNode)`: Adds `newNode` as a new (last) child node to `node`.

`node.insertBefore(newNode, oldNode)`: Inserts `newNode` as a new child node of `node` before `oldNode`.

`node.removeChild(oldNode)`: Removes the child `oldNode` from `node`.

`node.replaceChild(newNode, oldNode)`: Replaces the child node `oldNode` of `node` with `newNode`.

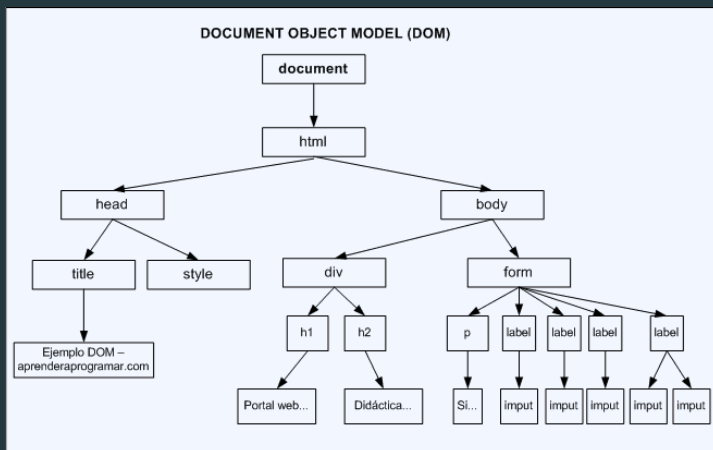
`element.innerHTML`: Reads or writes the HTML content of the given element as a string—including all child nodes with their attributes and text content.

Known browser quirks:

`getAttribute` and `setAttribute` are not reliable. Instead, assign the property of the element object directly: `obj.property = value`. Furthermore, some attributes are actually reserved words, so instead of `class` use `className` and instead of `for` use `HTMLfor`.

Real DOM compliant browsers will return linebreaks as text nodes in the `childNodes` collection, make sure to either remove them or test for the `nodeType`.

Resumen:



Estructurate



Localiza el
objetivo



Hazlo