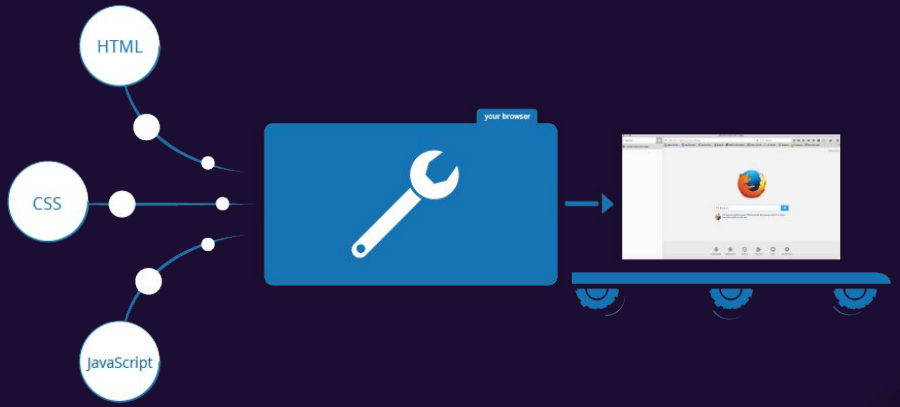
# Document Object Model
# DOM

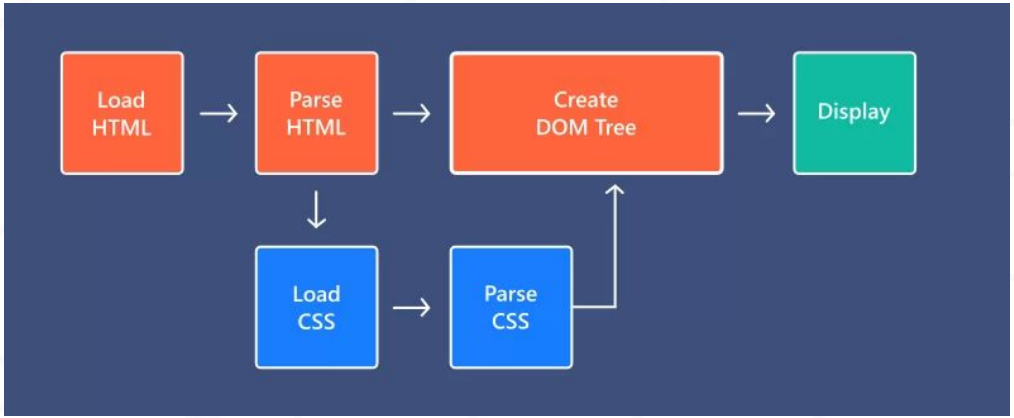
DEV.F::
DESARROLLAMOS(PERSONAS);

DEV.F

# *Carga de HTML / CSS y JS en el browser*



# ¿Cómo funciona?




Se recomienda que la etiqueta script cargue el JS al final del body, pos 2 principales razones:

- No detener el render.
- Qué el JS se cargue una vez pintada toda la parte visual

# DOM

DEV.F::
DESARROLLAMOS(PERSONAS);

# Document Object Model

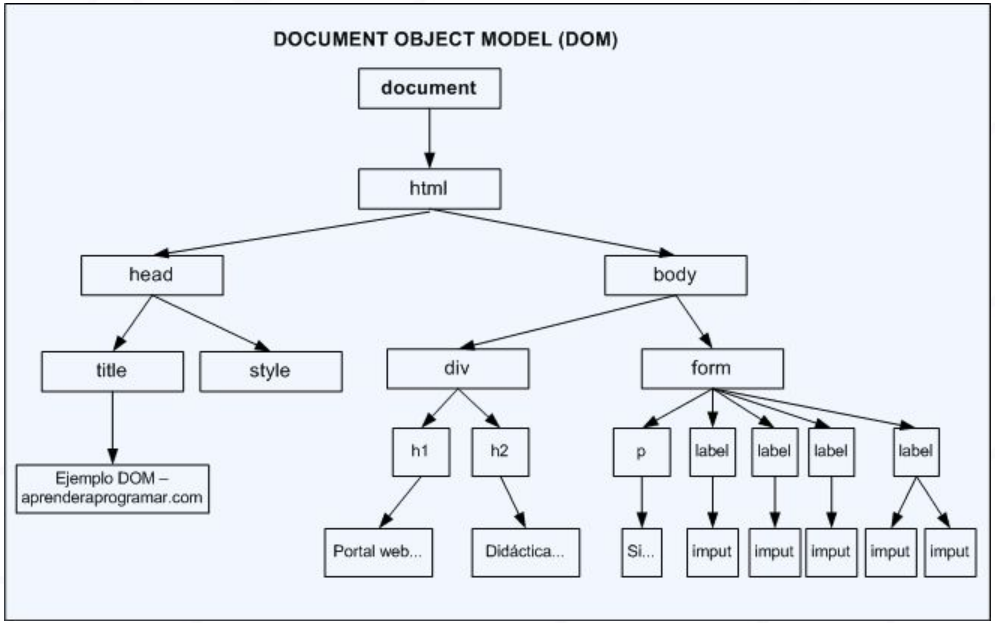
El **DOM** de un HTML es un **modelo de objetos** estándar y una **interfaz** de programación para HTML.

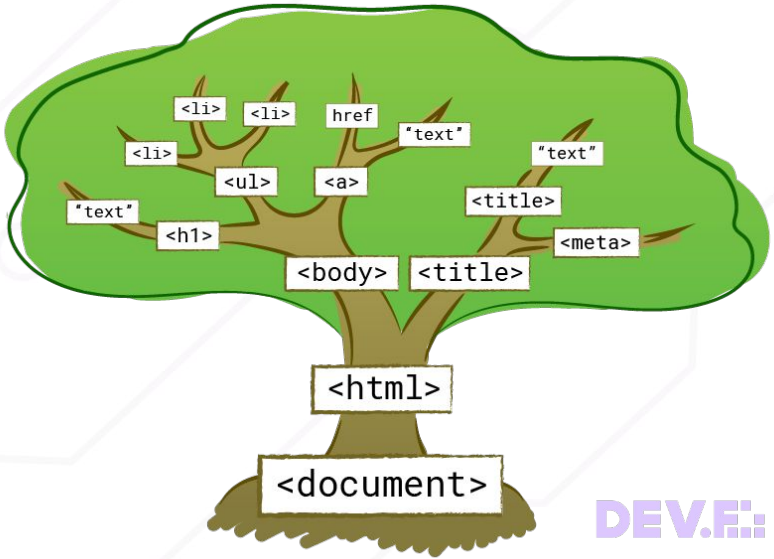
Se compone de:

- Los elementos **HTML** como **objetos.**
- Las **propiedades, métodos para acceder** y **eventos** de todos los elementos HTML.

El **DOM** de HTML **es el estándar** para cómo **obtener**, **modificar**, **cambiar** o **borrar** elementos HTML

# Document Object Model



# ¿Para qué sirve?

La interacción de una página web y los cambios en la interfaz gráfica son el centro de una web ya que son su funcionalidad, es decir, lo que le da valor al usuario. Para lograr esto, requerimos manipular el DOM.

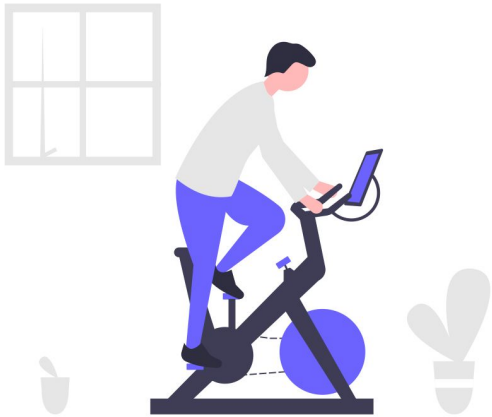
- Agregar dinámicas a nuestra web (crear/modificar elementos del DOM).
- Navegar entre páginas.
- Validar formularios.
- Consultar APIs.

# Modificando el DOM

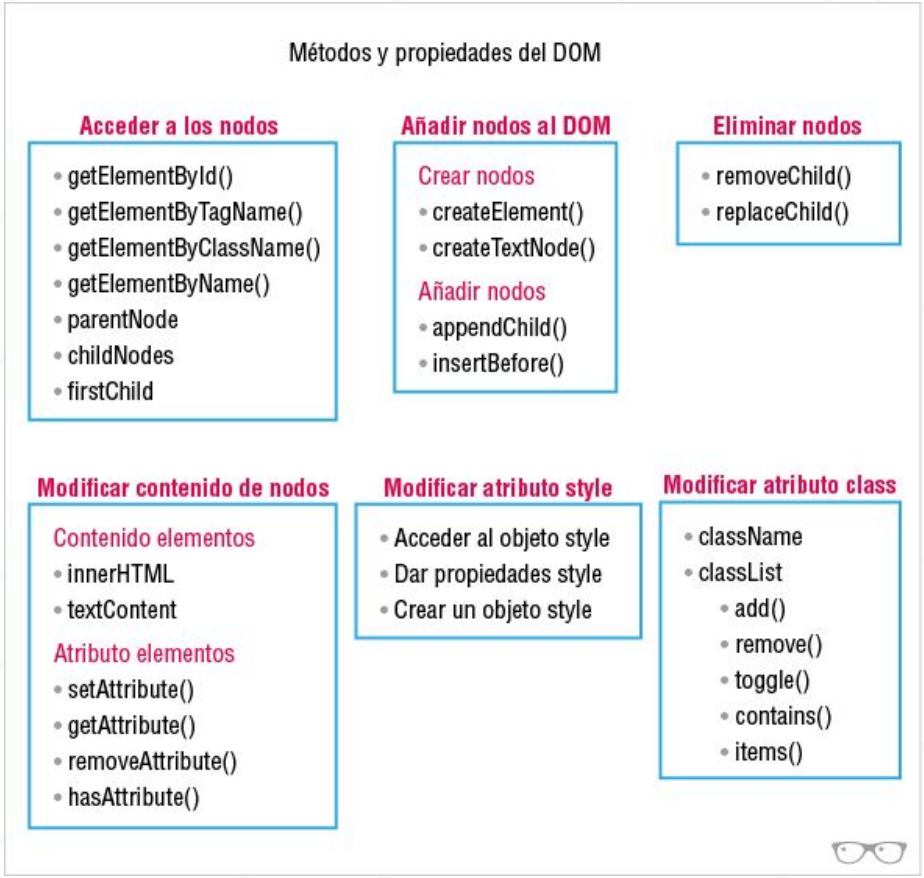
DEV.F:: DESARROLLAMOS(PERSONAS);

# Con JS se puede modificar el DOM

1. Leer los elementos.
2. Obtener valores, atributos y contenido de los elementos.
3. Crear nuevos elementos.
4. Eliminar elementos.
5. Agregar contenido, atributos, clases a los elementos.
6. **Volver dinámica nuestra web.**



# Métodos y propiedades del DOM

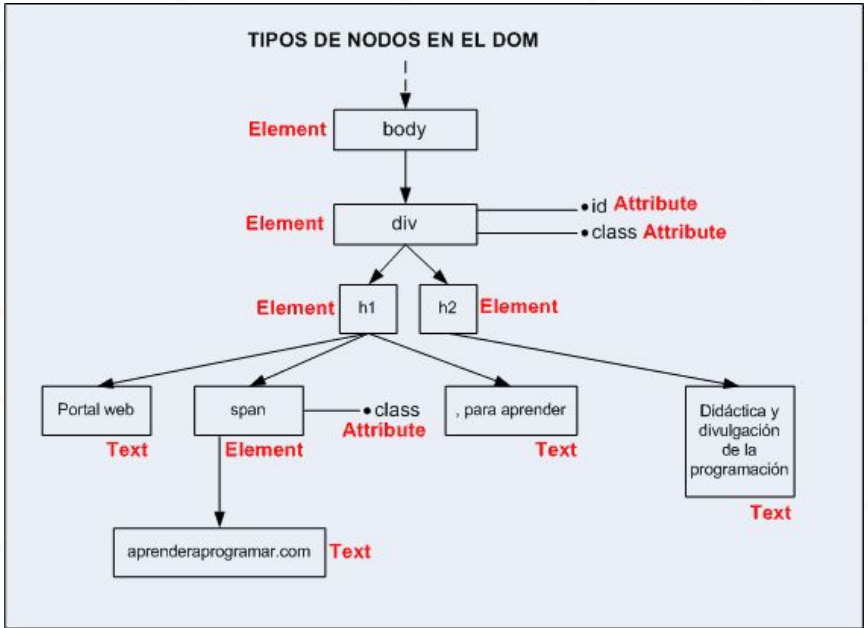


# Nodos y HTML List

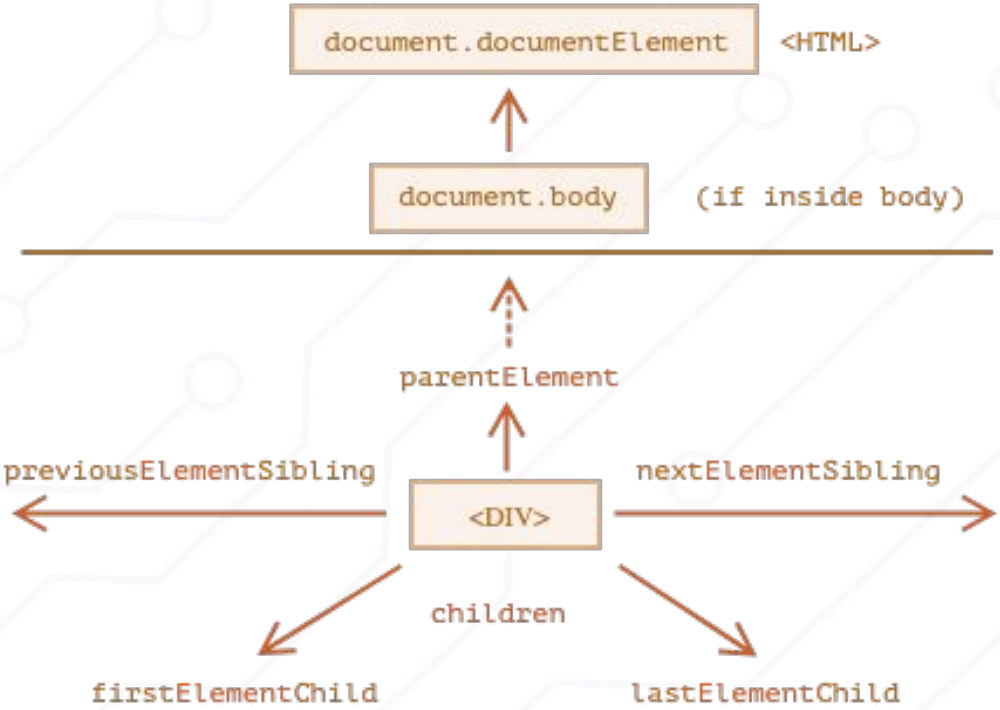
DEV.F::
DESARROLLAMOS(PERSONAS);

# Nodos

Es un objeto que devuelve un conjunto de nodos. Podríamos pensar que se parece a un array, por ser un conjunto de datos (Array: lista ordenada de datos) pero no lo es. De entrada, porque no podemos trabajar con él como si lo fuese.



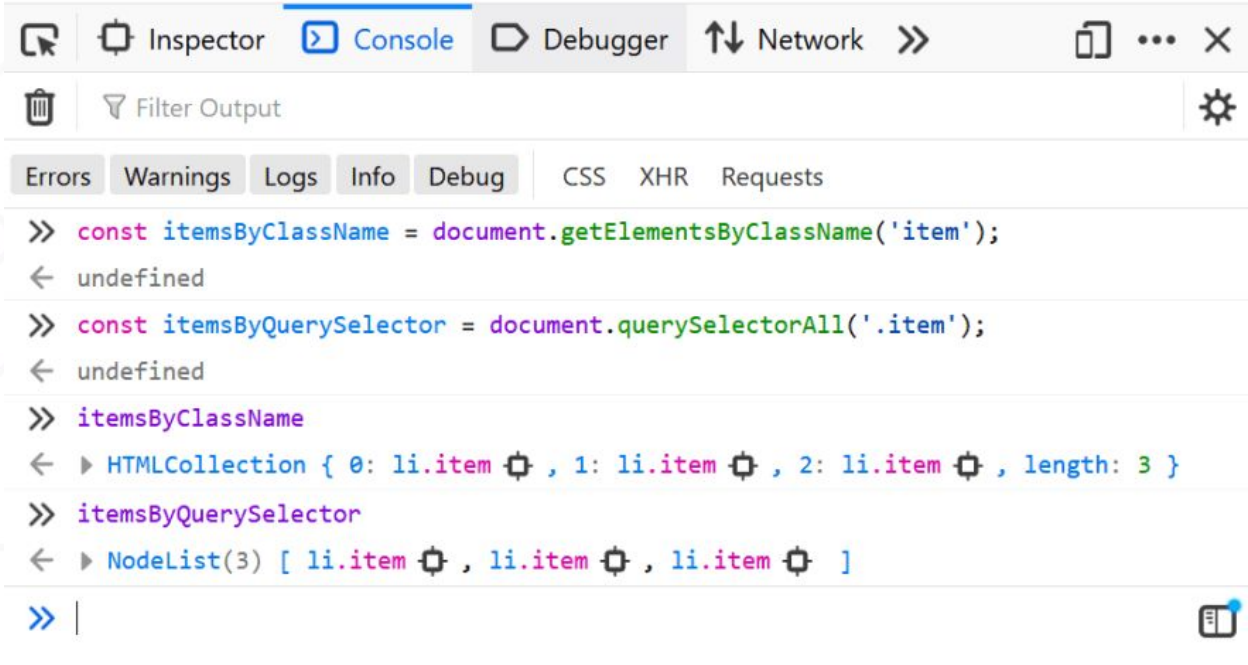
# Parent, siblings and children



# Node list

- Documentación.

- First Item
- Second Item
- Third Item



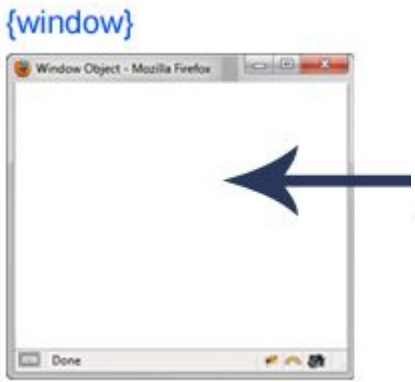
# Window y Document

DEV.F
DESARROLLAMOS(PERSONAS);

# window y document

window representa la **ventana que contiene** al **DOM**.

document es la propiedad que **apunta al DOM**, es decir es el objeto de **acceso al árbol html.**




# Eventos

DEV.F::
DESARROLLAMOS(PERSONAS);

# Eventos

Es una acción que ocurre sobre un elemento HTML, dicha acción es disparada por el usuario quien espera una respuesta. Esto se basa en el paradigma de programación orientada a eventos.

# Ejemplo de usar eventos HTML

Mediante **atributos HTML** se pueden **agregar eventos a los elementos HTML** y posteriormente escribir funciones en JavaScript que manejen las acciones que queremos que ocurran al suscitarse el evento.

- Lista de eventos en HTML.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Eventos HTML</title>
</head>
<body>
    <button onclick="sayHellow()">Fire event!</button>
    <script src="./index.js"></script>
</body>
</html>
```

```js
function sayHellow() {
    alert('hola mundo con onclick(html)...');
}
```

# Ejemplo de usar eventos JS

También podemos crear **event handlers** *(escuchadores/manejadores)* con JS para quitarle lógica al HTML. Estos escuchadores se agregan con **addEventListener** desde JS.

- Lista de eventos en JavaScript.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Eventos HTML</title>
</head>
<body>
    <button onclick="sayHellow()">Fire event!</button>
    <script src="./index.js"></script>
</body>
</html>
```

```js
const buttonElement = document.querySelector('#my-button');

buttonElement.addEventListener('click', function() {
    alert('hola mundo con event listener...');
})
```

# Objeto Event

El objeto Event representa un evento que tiene lugar en el DOM.

**Un evento** es disparado para **una acción del usuario** (click en un botón, scroll en una sección, presionar una tecla, etc.). Todo evento que se dispara permite capturar información sobre quien lo lanzó y una de las más importantes es **event.target**.

Doc de Event.

# jQuery

DEV.F::
DESARROLLAMOS(PERSONAS);

# jQuery

jQuery es una biblioteca de JavaScript que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, animaciones y agregar funciones AJAX.

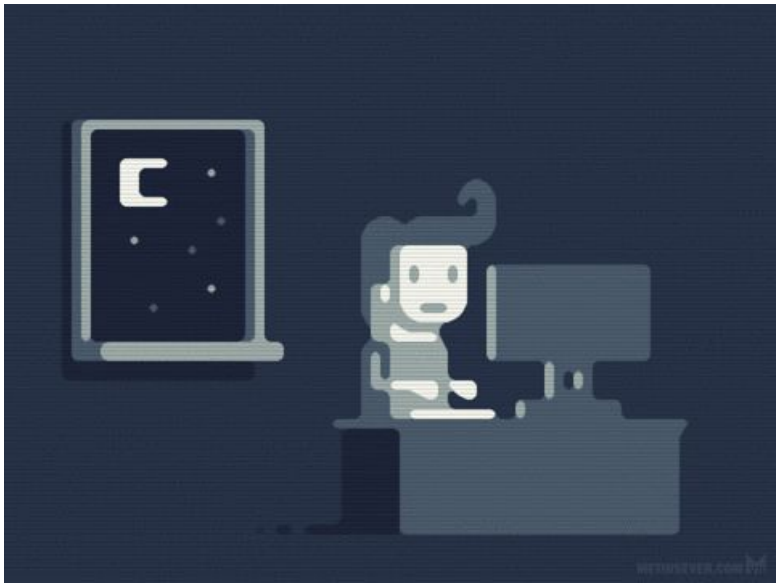
```
// js
document.getElementById("boton").style.display = 'none';


// jQuery

$("#boton").hide();
```

DEV.F

# Docs

DEV.F::
DESARROLLAMOS(PERSONAS);

# Documentación



- ¿Cómo se usa?
  - MDN.
  - W3Schools.
- ¿Puedo usarlo?
  - Can I Use
- Best practices
  - Code Guide html/css.
  - Clean code JS.

DEV.F::

DEV.F::

*Too much?*

# Consejos

1. Seleccionar la información que quieren adquirir.
2. Vivir una línea de código a la vez.
3. Tener más cosas que hacer y momentos de desconexión.
4. Trabajar en sí mismos y su inteligencia emocional.
5. Aceptar que no siempre es divertido.