Introducción a jQuery

1. ¿Qué es jQuery?

- **jQuery** es una biblioteca de JavaScript creada en 2006.
- Su lema: *"Write less, do more"* (escribe menos, haz más).
- Permite manipular el DOM, gestionar eventos, aplicar efectos y realizar peticiones AJAX de forma sencilla y compatible entre navegadores.

Ventajas:

- Sintaxis más corta y legible que JavaScript puro.
- Mayor compatibilidad entre navegadores antiguos.
- Gran comunidad y abundante documentación.

Inconvenientes:

- Hoy en día, muchos de sus usos se cubren con JavaScript moderno (ES6+).
- Puede ser una dependencia innecesaria en proyectos nuevos.

En resumen: jQuery sigue siendo útil para proyectos con compatibilidad amplia o mantenimiento de aplicaciones existentes.

---

2. Cómo usar jQuery

2.1. Inclusión en un proyecto

Podemos añadirlo en una página HTML de dos maneras:

a) Desde CDN (más común):

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

b) Descargando el archivo y enlazándolo:

<script src="js/jquery.min.js"></script>

---

3. Sintaxis básica

La estructura principal es:

$(selector).acción();

- $ → indica el uso de jQuery.
- selector → indica qué elemento del DOM seleccionamos (igual que CSS).

- acción() → método que aplicamos.

Ejemplo:

$("#miDiv").hide(); // Oculta el elemento con id="miDiv"

$(".rojo").css("color", "red"); // Cambia el color de todos los elementos con clase "rojo"

---

4. Selectores más comunes

- $("#id") → selecciona por id.
- $(".clase") → selecciona por clase.
- $("etiqueta") → selecciona por etiqueta HTML.
- $("div p") → selecciona todos los <p> dentro de un .
- $("p:first") → primer <p>.
- $("p:last") → último <p>.

---

5. Eventos en jQuery

Ejemplo con **click**:

$("#boton").click(function() {

 alert("¡Has hecho clic en el botón!");

});

Otros eventos:

- mouseenter / mouseleave
- keydown / keyup
- submit
- change

---

6. Manipulación del DOM

Cambiar contenido

$("#texto").text("Nuevo texto"); // Solo texto

```
$("#texto").html("<b>Texto en negrita</b>");  // HTML incluido
$("#caja").val("Nuevo valor");  // Para inputs
```

Añadir o eliminar elementos

```
$("#lista").append("<li>Nuevo elemento</li>");  // al final
$("#lista").prepend("<li>Nuevo elemento</li>"); // al inicio
$("#parrafo").remove();  // eliminar
```

---

7. Estilos y clases

```
$("#caja").css("background-color", "yellow");  // aplicar CSS directo
$("#caja").addClass("activo");   // añadir clase
$("#caja").removeClass("activo"); // eliminar clase
$("#caja").toggleClass("activo"); // alternar clase
```

---

8. Efectos y animaciones

```
$("#caja").hide();      // Ocultar
$("#caja").show();      // Mostrar
$("#caja").toggle();    // Alternar
$("#caja").fadeIn();    // Aparecer suavemente
$("#caja").fadeOut();   // Desaparecer suavemente
$("#caja").slideUp();   // Plegar hacia arriba
$("#caja").slideDown();  // Desplegar hacia abajo
```

---

9. AJAX con jQuery

Permite cargar datos sin recargar la página.

```
$.ajax({
  url: "datos.json",
  type: "GET",
  success: function(respuesta) {
```

```
  console.log(respuesta);
 },
 error: function() {
  alert("Error en la petición AJAX");
 }
});
```

Conclusión:

- jQuery simplifica la programación en JavaScript clásico.
- Es importante conocerlo porque todavía se usa en proyectos existentes.
- Sin embargo, en proyectos nuevos se recomienda usar **JavaScript moderno** y frameworks actuales (React, Vue, Angular).