INSTITUTO KHIPU

Semestre IV

Sesión 20

PROGRAMA DE ESTUDIOS

DESARROLLOS DE SISTEMAS DE INFORMACIÓN

DESARROLLO FRONTEND DE SISTEMAS DE WEB

Tema

INTRODUCCIÓN A AJAX CON JQUERY

Introducción a AJAX con jQuery

Objetivo de aprendizaje

Aprender a usar AJAX en jQuery para realizar peticiones asíncronas a servidores, manipular datos en formato JSON y actualizar páginas web dinámicamente *sin recargar*.

1. Teoría (¡En 3 pasos!)

¿Qué es AJAX?

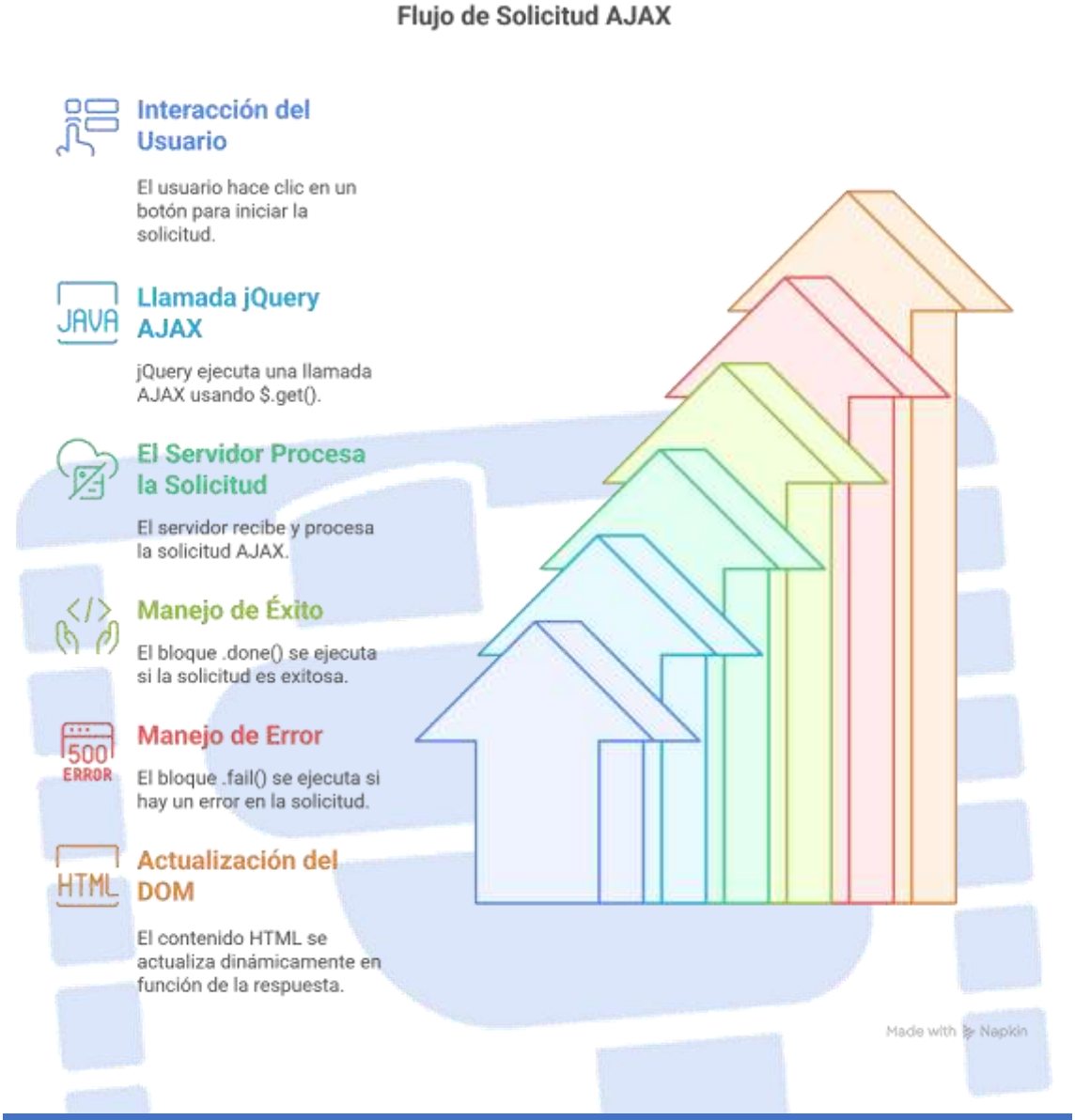
- **Qué**: Técnica para comunicar una página web con un servidor *en segundo plano* (sin recargar). Ejemplo:

Es como pedir comida por una app mientras realizas otra actividad (sigues viendo Netflix mientras llega).

- **Por qué**: Mejora la experiencia del usuario (ej.: cargar comentarios en redes sociales sin refrescar).
- **Cómo**: jQuery simplifica AJAX con métodos como `$.ajax()`, `$.get()` y `$.post()`.

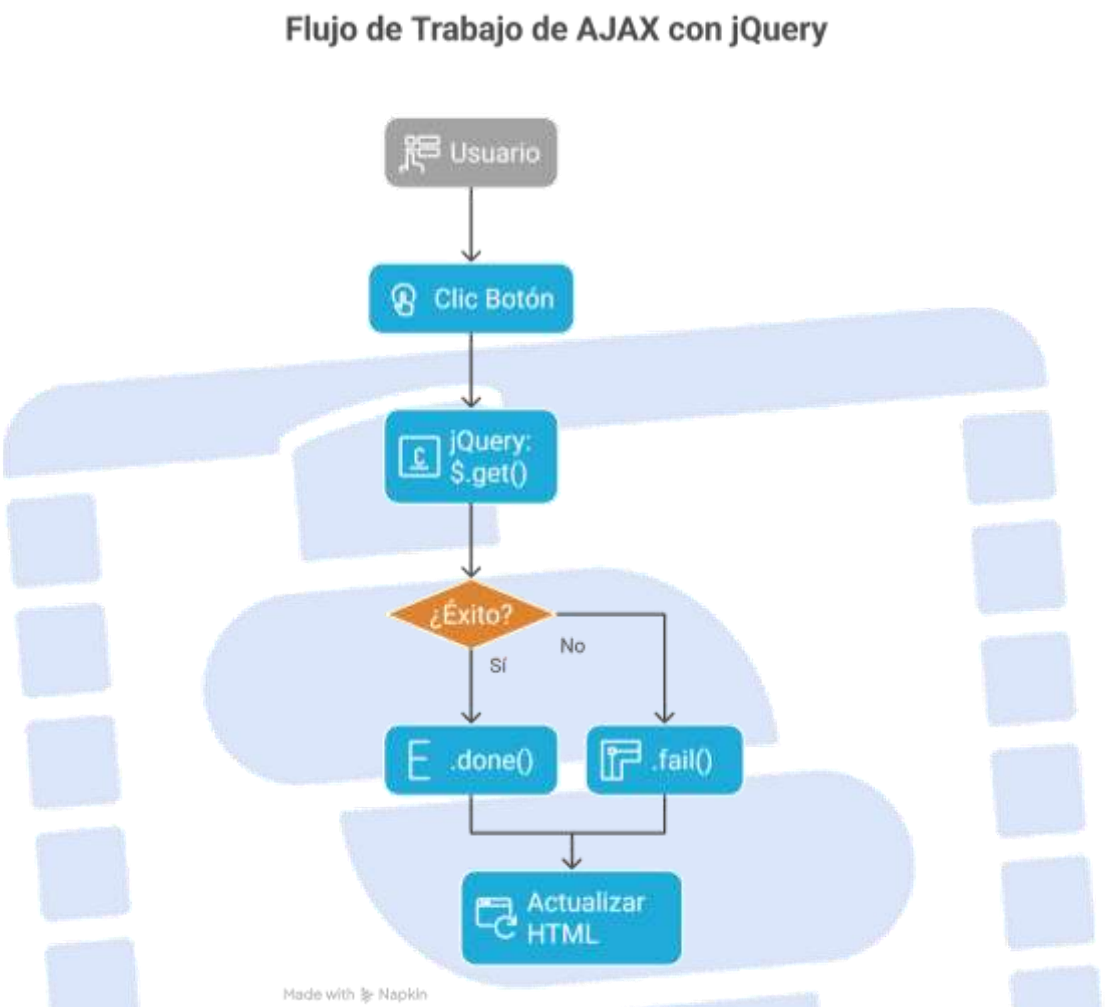
Métodos clave

Método	Uso	Ejemplo cotidiano
`$.ajax()`	Petición personalizada (GET/POST)	"Hazme un café, pero con 2 azúcares".
`$.get()`	Obtener datos (solo lectura)	"Dame el menú del día".
`$.post()`	Enviar datos al servidor	"Guarda mi pedido en la cocina".



Método	¿Cuándo usarlo?	Ejemplo Práctico
$.get()	Obtener datos (ej: API)	Cargar tweets en un feed.
$.post()	Enviar datos (ej: formulario)	Guardar un comentario en una red social.
$.ajax()	Personalizar toda la petición	Configurar headers o timeout.

A continuación, se muestra el flujo de cómo trabaja AJAX con JQuery:



Manejo de respuestas

```
$.get("https://jsonplaceholder.typicode.com/posts", function(data) {
  console.log(data); // Éxito
}).fail(function(error) {
  console.log("Error: " + error.status); // Falló
});
```

- **`done()`**: Éxito.
- **`fail()`**: Error.
- **`always()`**: Siempre se ejecuta (como "gracias" al final).

2. Ejercicio Guiado (Paso a Paso)

Objetivo: Crear un botón que cargue títulos de posts desde JSONPlaceholder.

Código completo:

```
<!DOCTYPE html>
<html>
<head>
  <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
</head>
<body>
  <button id="loadPosts">Cargar Posts</button>
  <div id="postList"></div>

  <script>
    $("#loadPosts").click(function() {
      $.get("https://jsonplaceholder.typicode.com/posts",
function(data) {
        let html = "";
        data.slice(0, 5).forEach(post => { // Solo primeros 5 posts
          html += `<p>${post.title}</p>`;
        });
        $("#postList").html(html);
      }).fail(function() {
        alert("¡Error al cargar los datos!");
      });
    });
  </script>
</body>
</html>
```

Pasos:

- Vincular jQuery.
- Escuchar clic en el botón.
- Hacer petición GET a la API.
- Renderizar títulos en el div `#postList`.

3. Desafío Independiente

Objetivo: Crear un formulario que envíe un nuevo post a JSONPlaceholder (simulado) usando `$.post()`.

Requisitos:

- Campos: `title` y body.
- Mostrar "Post enviado" al recibir respuesta.

  API falsa (no guarda datos realmente):

```
$.post("https://jsonplaceholder.typicode.com/posts", {
  title: "Mi título",
  body: "Mi contenido"
});
```

Retroalimentación sugerida:

- ¿Manejaste el estado de "cargando"? (ej.: `$("#form").text("Enviando...")`).
- ¿Validaste campos vacíos?

4. Preguntas de Autoevaluación

- ¿Por qué usar AJAX en lugar de un formulario tradicional?
- ¿Qué diferencia hay entre `$.get()` y `$.post()`?
- ¿Cómo manejarías un error de conexión en una petición?

Nota: Para la tarea, usa la API JSONPlaceholder para crear una tabla dinámica con posts.

INSTITUTO

KHIPU