INSTITUTO KHIPU

Semestre IV

Sesión 21

PROGRAMA DE ESTUDIOS

# DESARROLLOS DE SISTEMAS DE INFORMACIÓN

## DESARROLLO FRONTEND DE SISTEMAS DE WEB

Tema

## PETICIONES AJAX CON JQUERY (.GET() Y .POST())

# Peticiones AJAX con jQuery (.get() y .post())

**Objetivo de Aprendizaje**

Dominar el uso de `.get()` y `.post()` en jQuery para realizar peticiones asíncronas a servidores, enviar/recibir datos sin recargar la página y manejar respuestas del servidor para actualizar el DOM.

## Sección 1: Teoría

## ¿Qué es AJAX?

AJAX (Asynchronous JavaScript and XML) permite actualizar partes de una página web sin recargarla completamente. Con jQuery, AJAX se vuelve más sencillo gracias a sus métodos abreviados como `.get()` y `.post()`.

## ¿Por qué se usa AJAX?

- Mejora la **experiencia del usuario** al evitar recargas completas.
- Permite **interacciones más dinámicas** entre cliente y servidor.
- Facilita el consumo de **APIs públicas o privadas**.

### ¿Por qué AJAX con jQuery?

AJAX permite comunicación cliente-servidor en segundo plano. Por ejemplo, como actualizar el saldo de tu banca móvil sin salir de la app. jQuery simplifica el código comparado con JavaScript puro.

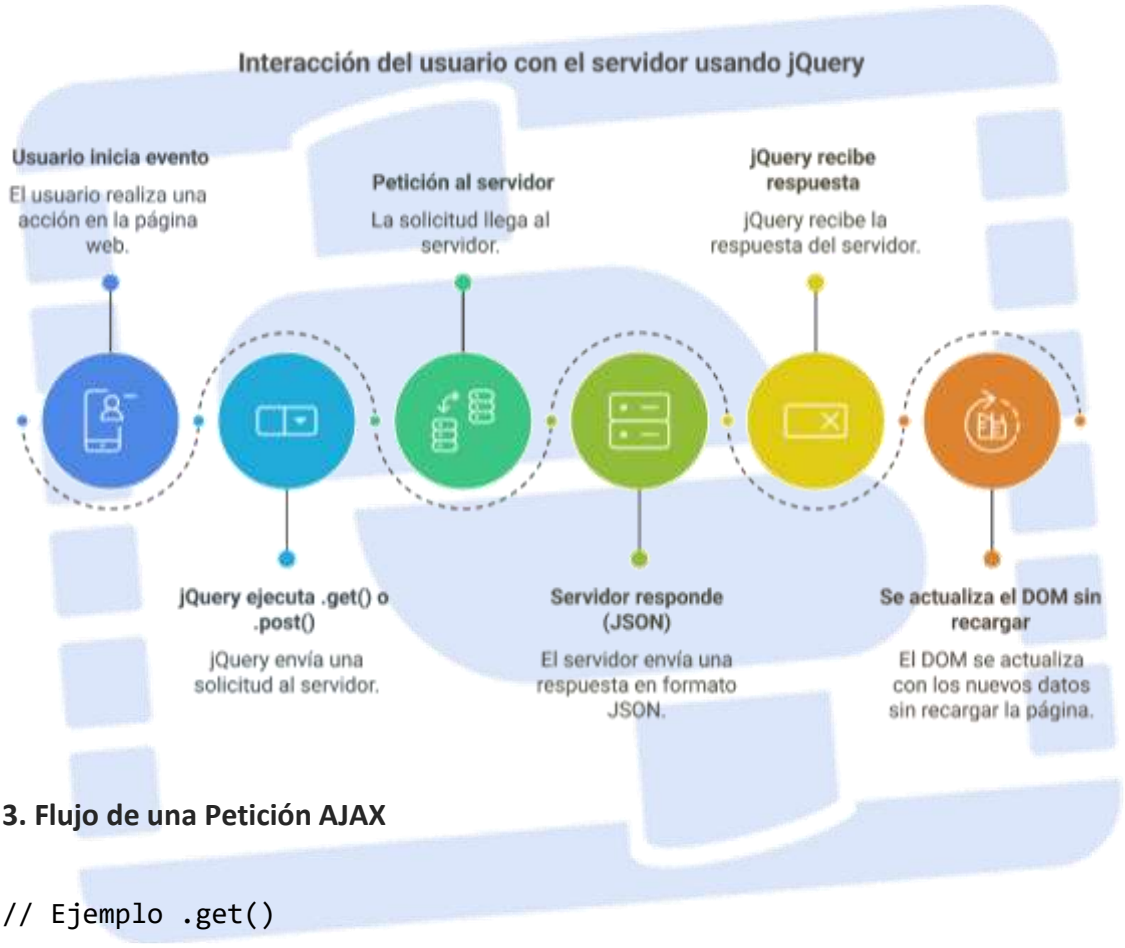
## Métodos AJAX en jQuery

- **`.get(url, data, callback)`**
- Realiza una petición HTTP GET.
- Ideal para recuperar datos (lectura).
- **`.post(url, data, callback)`**
- Realiza una petición HTTP POST.

- Ideal para enviar datos al servidor (creación).

### 2. Diferencias Clave: `.get()` vs `.post()`

| Método | Uso Principal | Ejemplo Real |
|---|---|---|
| `.get()` | Obtener datos (lectura) | Cargar productos de un catálogo. |
| `.post()` | Enviar datos (escritura) | Enviar un formulario de contacto. |




### 3. Flujo de una Petición AJAX

```
// Ejemplo .get()
$.get("https://api.example.com/data", function(respuesta) {
  console.log("Datos recibidos:", respuesta);
});

// Ejemplo .post()
$.post("https://api.example.com/guardar", { nombre: "Juan" },
function(respuesta) {
```

```
  console.log("Respuesta del servidor:", respuesta);
});
```

## Sección 2: Ejercicio Guiado (Paso a Paso)

### Demostración 1: `.get()` – Recuperar datos de una API

```
<button id="cargar">Cargar usuarios</button>
<table id="usuarios"></table>

<script>
 $("#cargar").click(function() {
  $.get("https://jsonplaceholder.typicode.com/users", function(data) {
   let tabla = "<tr><th>Nombre</th><th>Email</th></tr>";
   data.forEach(user => {
    tabla += `<tr><td>${user.name}</td><td>${user.email}</td></tr>`;
   });
   $("#usuarios").html(tabla);
  });
 });
</script>
```

### Demostración 2: `.post()` – Enviar datos al servidor

```
<form id="formulario">
 <input type="text" name="nombre" placeholder="Nombre">
 <button type="submit">Enviar</button>
</form>

<script>
 $("#formulario").submit(function(e) {
  e.preventDefault();
  const datos = $(this).serialize();
  $.post("https://jsonplaceholder.typicode.com/posts", datos, function(respuesta) {
   alert("Enviado. ID generado: " + respuesta.id);
  });
 });
</script>
```

**Sección 3: Actividad práctica individual**

## Instrucciones:

Crea una página web que:

- Al hacer clic en un botón, use `.get()` para obtener usuarios desde una API pública y los muestre en una tabla.
- Tenga un formulario con nombre y correo, que al enviarse, use `.post()` para enviar los datos y muestre una alerta con la confirmación.

## Recursos sugeridos:

- API de prueba: https://jsonplaceholder.typicode.com/users (GET)
- API de prueba: https://jsonplaceholder.typicode.com/posts (POST)

## Sección 4: Preguntas de Autoevaluación

- ¿Cuándo usarías `.post()` en lugar de `.get()`?
- ¿Cómo evitarías que un usuario envíe datos duplicados?
- ¿Por qué es importante usar `e.preventDefault()` en formularios con AJAX?

## Tarea para Casa

**Aplicación de Clima**:

- Usar `.get()` con la API OpenWeatherMap
- Mostrar temperatura y condiciones en una tarjeta
- Extra: Añadir un botón para buscar por ciudad

## Pregunta reflexiva:

¿Cuándo es más adecuado usar `.get()` y cuándo `.post()`?

- `.get()` → leer información.

- `.post()` → enviar o crear recursos.

## 🗄 Tarea:

Desarrollar una aplicación sencilla que:

- Use `.get()` para mostrar una lista de publicaciones desde una API pública.
- Permita enviar comentarios mediante `.post()`.

INSTITUTO

KHIPU