

## Diferencia entre un POST clásico con formulario y un POST con fetch.

### 1. POST clásico con formulario

Al hacer clic en **Enviar POST (formulario)**:

- En **Network** aparecerá una petición nueva, tipo **document** (porque abre otra página o recarga).
- Si hacen clic en la petición → pestaña **Headers**:
  - Request Method: POST.

Name	×	Headers	Payload	Preview	Response	Initiator	Timing
post		▼ General					
		Request URL	https://httpbin.org/post				
		Request Method	POST				
		Status Code	200 OK				
		Remote Address	52.71.132.100:443				
		Referrer Policy	strict-origin-when-cross-origin				
		▼ Response Headers					

- En **Request Headers**, el navegador añade automáticamente Content-Type: application/x-www-form-urlencoded.

Name	×	Headers	Payload	Preview	Response	Initiator	Timing
post		▼ Response Headers					
		Access-Control-Allow-Credentials	true				
		Access-Control-Allow-Origin	null				
		Content-Length	1231				
		Content-Type	application/json				
		Date	Tue, 16 Sep 2025 18:02:27 GMT				
		Server	unicorn/19.9.0				
		▼ Request Headers					
		:authority	httpbin.org				
		:method	POST				
		:path	/post				
		:scheme	https				
		Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7				
		Accept-Encoding	gzip, deflate, br, zstd				
		Accept-Language	es-ES,es;q=0.9				
		Cache-Control	max-age=0				
		Content-Length	24				
		Content-Type	application/x-www-form-urlencoded				
		Origin	null				

- En Payload, en **Form Data** se verá algo así:  
usuario=admin&clave=1234

Name	×	Headers	Payload	Preview	Response	Initiator	Timing
post		▼ Form Data	View source	View URL-encoded			
		usuario	admin				
		clave	1234				

- En **Response**, se verá la respuesta completa del servidor

```

1  {
2    "args": {},
3    "data": "",
4    "files": {},
5    "form": {
6      "clave": "1234",
7      "usuario": "admin"
8    },
9    "headers": {
10     "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8",
11     "Accept-Encoding": "gzip, deflate, br, zstd",
12     "Accept-Language": "es-ES,es;q=0.9",
13     "Cache-Control": "max-age=0",
14     "Content-Length": "24",
15     "Content-Type": "application/x-www-form-urlencoded",
16     "Host": "httpbin.org",
17     "Origin": "null",
18     "Priority": "u=0, i",
19     "Sec-Ch-Ua": "\"Not;A=Brand\";v=\"99\", \"Google Chrome\";v=\"139\", \"Chromium\";v=\"139\"",
20     "Sec-Ch-Ua-Mobile": "?1",
21     "Sec-Ch-Ua-Platform": "\"Android\"",
22     "Sec-Fetch-Dest": "document",
23     "Sec-Fetch-Mode": "navigate",
24     "Sec-Fetch-Site": "cross-site",
25     "Sec-Fetch-User": "?1",
26     "Upgrade-Insecure-Requests": "1",
27     "User-Agent": "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko)",
28     "X-Amzn-Trace-Id": "Root=1-68c9a632-4bf0429f561b2922188bf04d"
29   },
30   "json": null,
31   "origin": "85.235.68.121",
32   "url": "https://httpbin.org/post"
33 }
34

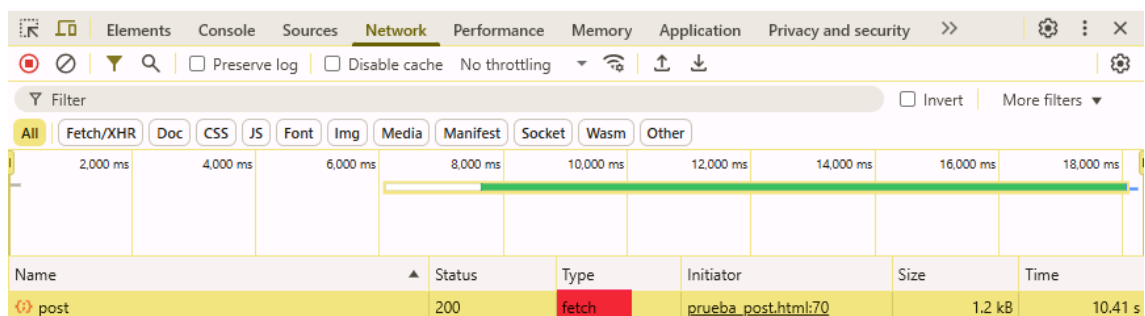
```

👉 Reseñable: el navegador **recarga o abre otra pestaña** → el POST va “pegado” a la navegación.

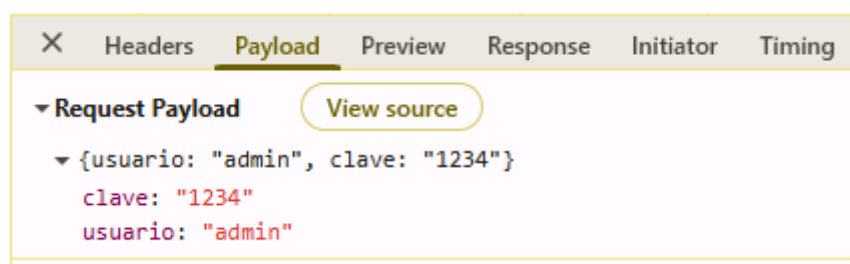
## ◆ 2. POST con fetch (AJAX/JS)

Cuando hagan clic en **Enviar POST con fetch()**:

- En **Network**, aparecerá una petición tipo **fetch / xhr** (según Chrome).



- Si la seleccionan → pestaña **Headers**:
  - Verán Request Method: POST.
  - Verán el **Request Payload** en JSON:



- En **Request Headers**, notarán Content-Type: application/json (lo pusimos nosotros en fetch).

▼ Request Headers	
:authority	httpbin.org
:method	POST
:path	/post
:scheme	https
Accept	*/*
Accept-Encoding	gzip, deflate, br, zstd
Accept-Language	es-ES,es;q=0.9
Content-Length	34
Content-Type	application/json

- En **Preview / Response**, verán la respuesta JSON que httpbin devuelve.

X	Headers	Payload	Preview	Response	Initiator	Timing
▼	<pre>{args: {}, data: '{"usuario":"admin","clave":"1234"}', files: {}, form: {},...}   args: {}   data: '{"usuario":"admin","clave":"1234"}'   files: {}   form: {}   headers: {Accept: "*/*", Accept-Encoding: "gzip, deflate, br, zstd", Accept-Language: "es-ES,es;q=0.9",...}     Accept: "*/*"     Accept-Encoding: "gzip, deflate, br, zstd"     Accept-Language: "es-ES,es;q=0.9"     Content-Length: "34"     Content-Type: "application/json"     Host: "httpbin.org"     Origin: "null"     Priority: "u=1, i"     Sec-Ch-Ua: "\"Not;A=Brand\";v=\"99\", \"Google Chrome\";v=\"139\", \"Chromium\";v=\"139\""     Sec-Ch-Ua-Mobile: "?1"     Sec-Ch-Ua-Platform: "\"Android\""     Sec-Fetch-Dest: "empty"     Sec-Fetch-Mode: "cors"     Sec-Fetch-Site: "cross-site"     User-Agent: "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) X-Amzn-Trace-Id: \"Root=1-68c9a9f6-482439e310a335710f39a5e8\""   json: {clave: "1234", usuario: "admin"}     clave: "1234"     usuario: "admin"     origin: "85.235.68.121"     url: "https://httpbin.org/post"   }</pre>					

```
X Headers Payload Preview Response Initiator Timing
1 {
2   "args": {},
3   "data": "{\"usuario\":\"admin\",\"clave\":\"1234\"}",
4   "files": {},
5   "form": {},
6   "headers": {
7     "Accept": "*/*",
8     "Accept-Encoding": "gzip, deflate, br, zstd",
9     "Accept-Language": "es-ES,es;q=0.9",
10    "Content-Length": "34",
11    "Content-Type": "application/json",
12    "Host": "httpbin.org",
13    "Origin": "null",
14    "Priority": "u=1, i",
15    "Sec-Ch-Ua": "\"Not;A=Brand\";v=\"99\", \"Google Chrome\";v=\"139\", \"Chromium\";v=\"139\"",
16    "Sec-Ch-Ua-Mobile": "?1",
17    "Sec-Ch-Ua-Platform": "\"Android\"",
18    "Sec-Fetch-Dest": "empty",
19    "Sec-Fetch-Mode": "cors",
20    "Sec-Fetch-Site": "cross-site",
21    "User-Agent": "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML,
22    "X-Amzn-Trace-Id": "Root=1-68c9a9f6-482439e310a335710f39a5e8"
23  },
24  "json": {
25    "clave": "1234",
26    "usuario": "admin"
27  },
28  "origin": "85.235.68.121",
```

- Pero (a diferencia del formulario), **la página no se recarga**: los datos se capturan en el JS y se pintan en el <pre> de la web.

👉 Reseñable: el POST se hace “en segundo plano”, la web sigue igual y el **resultado lo decide el programador en JS**.

---

## ✅ Resumen para remarcar en clase

### 1. Ambos son POST, pero:

- Formulario → Content-Type: application/x-www-form-urlencoded, recarga la página.
  - fetch → Content-Type: application/json, no recarga, controla la respuesta en JS.
-