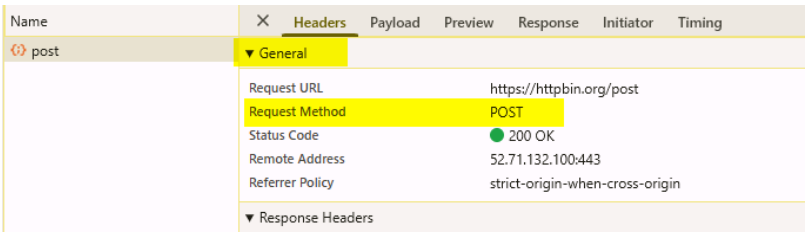
D**iferencia entre un POST clásico con formulario y un POST con fetch**.

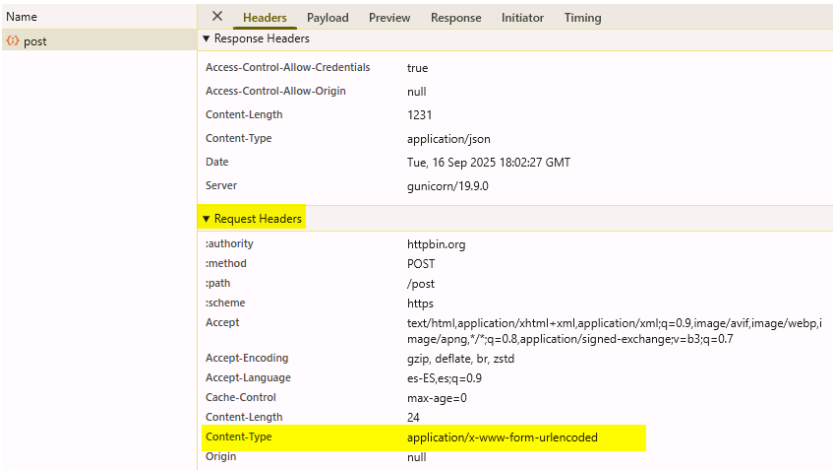
1. POST clásico con formulario

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

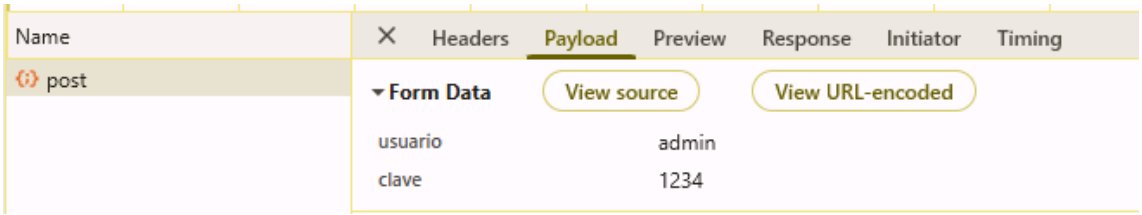
- En **Network** aparecerá una petición nueva, tipo **document** (porque abre otra página o recarga).
- Si hacen clic en la petición → pestaña **Headers**:
    - Request Method: POST.



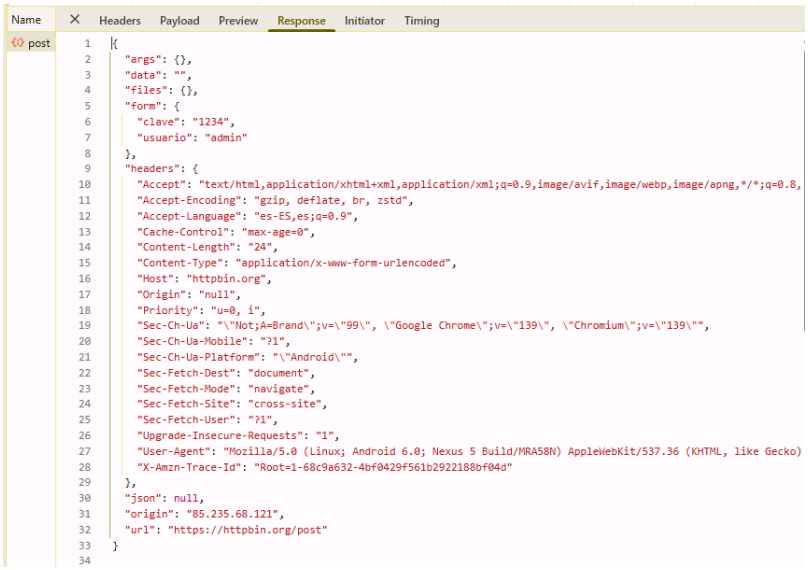
    - En **Request Headers**, el navegador añade automáticamente Content-Type: application/x-www-form-urlencoded.



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



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

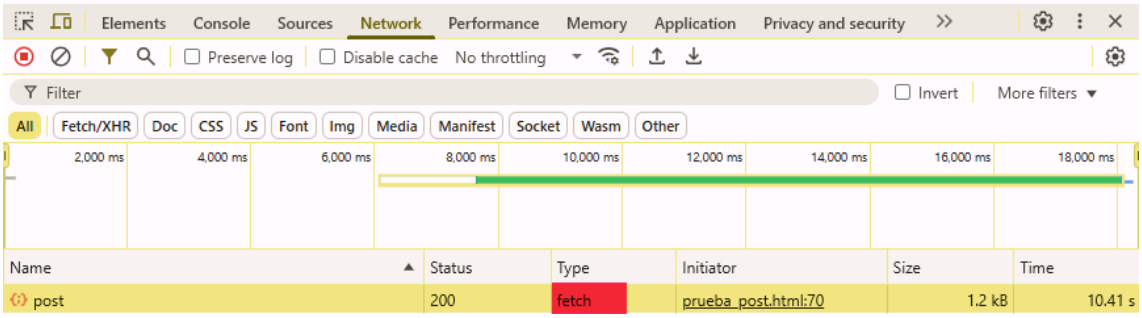


👉 Reseñable: el navegador **recarga o abre otra pestaña** → el POST va "pegado" a la navegación.

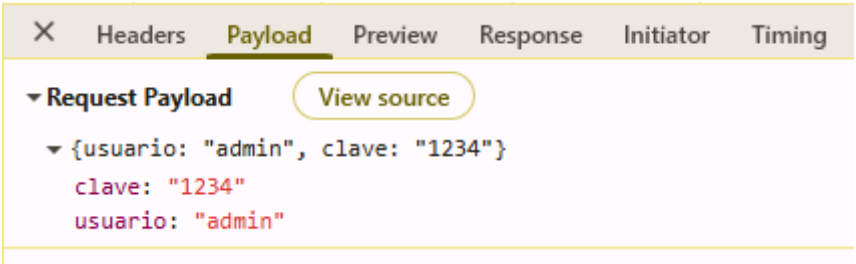
🔹 2. POST con fetch (AJAX/JS)

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

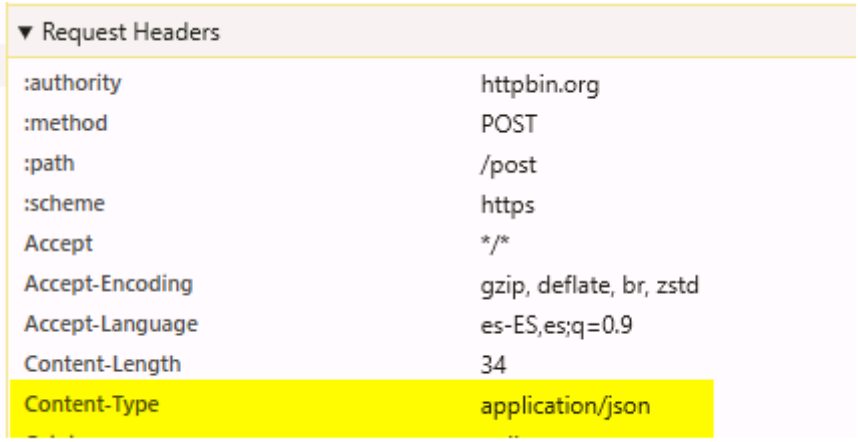
- En **Network**, aparecerá una petición tipo **fetch / xhr** (según Chrome).



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



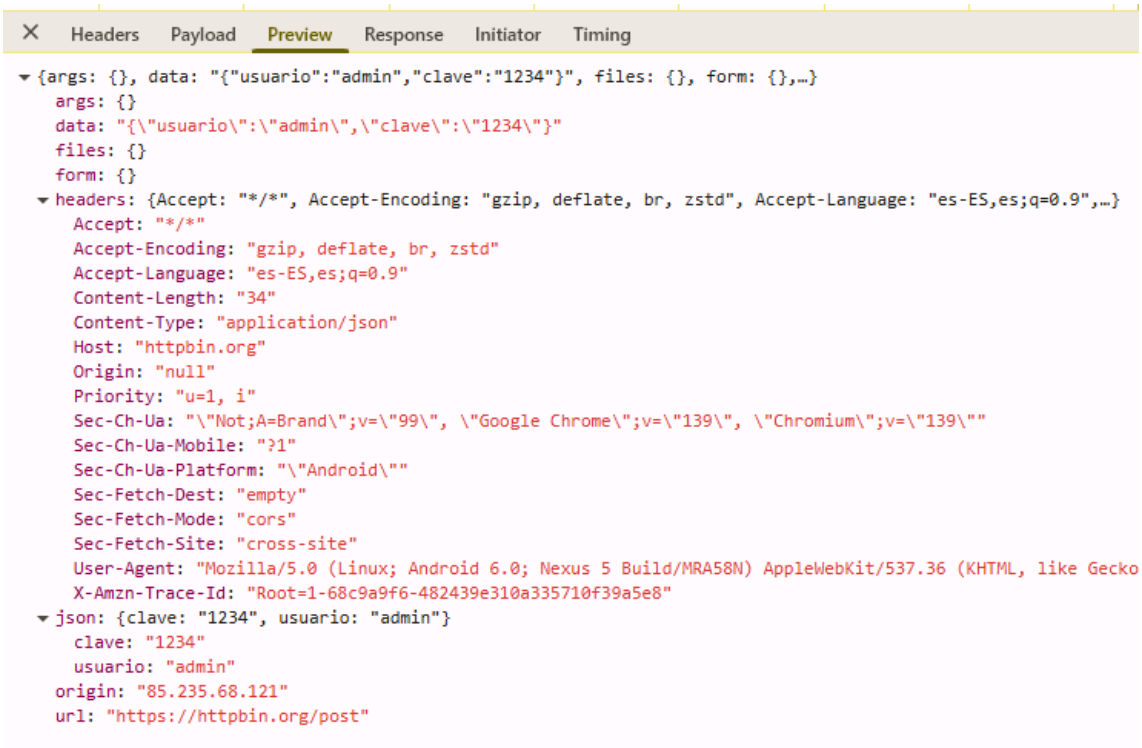
- En **Request Headers**, notarán Content-Type: application/json (lo pusimos nosotros en fetch).



▼ Request Headers

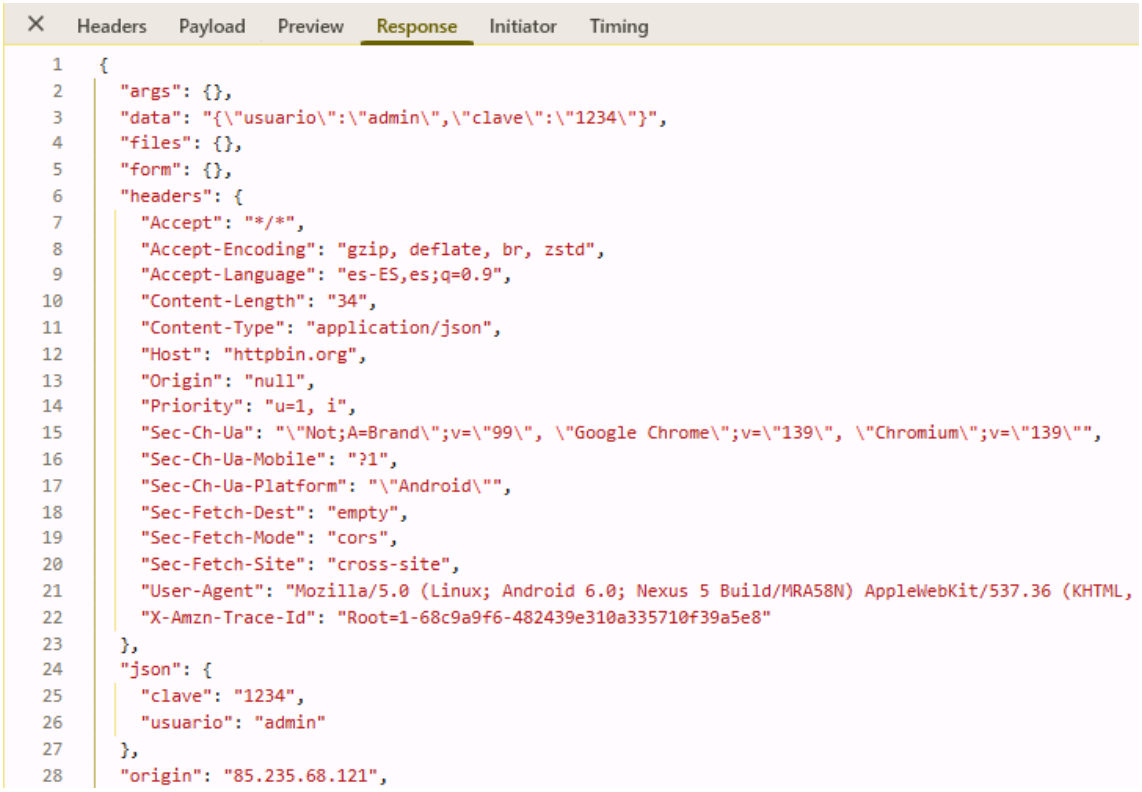
:authority	httpbin.org
:method	POST
:path	/post
:scheme	https
Accept	*/*
Accept-Encoding	gzip, deflate, br, zstd
Accept-Language	es-ES,es;q=0.9
Content-Length	34
Content-Type	application/json

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



Headers Payload Preview Response Initiator Timing


```
▼ {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"
```



- 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.
