Materia:

DISEÑO ELECTRÓNICO BASADO EN SISTEMAS EMBEBIDOS

Alumno:

Posadas Pérez Isaac Sayeg

Paniagua Rico Juan Julian

García Azzúa Jorge Roberto

Grado y grupo:

8°G

Profesor:

Garcia Ruiz Alejandro Humberto

Unidad 4 -Tarea 11:

Peticiones HTTP y códigos

Peticiones HTTP y Códigos de Estado

Introducción

El **protocolo HTTP (Hypertext Transfer Protocol)** es el estándar utilizado para la comunicación entre clientes (como navegadores o aplicaciones) y servidores en la web. Su funcionamiento se basa en **peticiones** realizadas por el cliente y **respuestas** emitidas por el servidor. Cada interacción incluye una **petición HTTP** con un método específico y una **respuesta** que contiene un **código de estado** que indica el resultado de la operación. Comprender estos elementos es esencial para trabajar con APIs y desarrollar aplicaciones web o móviles eficientes.

Desarrollo

Métodos de Petición HTTP

Los métodos o **verbos HTTP** definen la acción que se desea realizar sobre un recurso. Los más comunes son:

- **GET:** Solicita la recuperación de información de un recurso.
- **POST:** Envía datos al servidor para crear un nuevo recurso.
- **PUT:** Reemplaza un recurso existente con nuevos datos.
- **PATCH:** Modifica parcialmente un recurso.
- **DELETE:** Elimina un recurso.
- **OPTIONS:** Solicita información sobre las opciones de comunicación disponibles para un recurso.
- **HEAD:** Igual que GET, pero sin el cuerpo de la respuesta; se usa para obtener solo los encabezados.

Cada método tiene un propósito definido y debe utilizarse adecuadamente para cumplir con los principios RESTful y mantener la semántica de la API.

Códigos de Estado HTTP

Los **códigos de estado HTTP** son números de tres cifras que indican el resultado de una petición. Se agrupan en cinco clases principales:

Clase	Significado	Ejemplo
1xx	Informativos	100 Continue
2xx	Éxito	200 OK, 201 Created
3xx	Redirección	301 Moved Permanently, 304 Not Modified
4xx	Error del cliente	400 Bad Request, 401 Unauthorized, 404 Not Found
5xx	Error del servidor	500 Internal Server Error, 503 Service Unavailable

Códigos comunes explicados:

- **200 OK:** La petición fue exitosa.
- **201 Created:** El recurso fue creado correctamente (usualmente con POST).
- **204 No Content:** La operación fue exitosa pero no hay contenido que retornar.
- **400 Bad Request:** La petición es incorrecta o malformada.
- **401 Unauthorized:** No se ha proporcionado una autenticación válida.
- **403 Forbidden:** El cliente está autenticado, pero no tiene permisos.
- **404 Not Found:** El recurso solicitado no existe.

- **500 Internal Server Error:** Error genérico del servidor.
- **503 Service Unavailable:** El servidor está temporalmente fuera de servicio.

Estos códigos permiten diagnosticar el comportamiento de una API y aplicar manejo de errores eficiente desde el cliente.

Ejemplo práctico

Supongamos que una aplicación cliente realiza la siguiente solicitud:

```
GET /usuarios/10 HTTP/1.1
Host: api.ejemplo.com
Authorization: Bearer abc123
```

El servidor podría responder con:

```
HTTP/1.1 200 OK
Content-Type: application/json

{
  "id": 10,
  "nombre": "Laura Gómez",
  "email": "laura@example.com"
}
```

```
```

Pero si el ID no existe, el servidor puede devolver:

```
HTTP/1.1 404 Not Found
Content-Type: application/json

{
  "error": "Usuario no encontrado"
}
```

Y si no se proporciona el token de autenticación:

```
HTTP/1.1 401 Unauthorized
```

Estos códigos ayudan tanto al desarrollador como al sistema cliente a entender cómo manejar la respuesta.

Conclusión

Las **peticiones HTTP** y sus correspondientes **códigos de estado** son fundamentales en la interacción entre clientes y servidores web. Entender el propósito de cada método y la interpretación correcta de los códigos de respuesta permite diseñar sistemas más robustos, seguros y eficientes. En el contexto del desarrollo de APIs, este conocimiento es esencial para una comunicación clara entre servicios y para una depuración efectiva de errores.

Bibliografía

1. Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*.

2. Mozilla Developer Network (MDN). (2024). *HTTP Methods & Status Codes*. https://developer.mozilla.org

3. RFC 7231. *Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. IETF.

4. Postman Learning Center. (2023). *Working with HTTP Requests*. [https://learning.postman.com]

5. Microsoft Docs. (2023). *HTTP Status Codes*. [https://learn.microsoft.com]