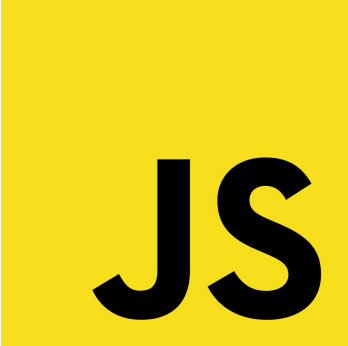


Introducción a Javascript

Parte 3

A yellow square containing the letters 'JS' in a bold, black, sans-serif font, representing the JavaScript logo.

JS

Repaso HTTP

- Es un protocolo.
- Sigue el modelo cliente-servidor.
- Es de texto plano.
- Sin estado.
- Dos tipos de mensajes:
 - Request (solicitud)
 - Método, URI, versión de HTTP, encabezado (headers), body (opcional)
 - Response (respuesta)
 - Versión de HTTP, Código de estado, mensaje, headers, body (opcional)

Repaso HTTP

- Métodos, que especifican la acción que el cliente quiere realizar sobre un recurso.
 - GET
 - POST
 - PUT
 - DELETE
 - HEAD
 - OPTIONS
 - PATCH

Repaso HTTP

- Códigos de estado
 - 1xx -> procesando solicitud.
 - 2xx -> éxito
 - 3xx -> redirección
 - 4xx -> error del cliente
 - 5xx -> error del servidor

<https://www.dotcom-monitor.com/wiki/es/knowledge-base/http-status-codes/>

Repaso HTTP

- Headers
 - Brindan información adicional de la solicitud o respuesta.
 - Algunos ejemplos son:
 - Content-Type (text/html, application/pdf, image/png, application/json)
 - Content-Length (en bytes)
 - User-Agent (ejemplo, Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36 Edg/91.0.864.59)
 - <https://developer.mozilla.org/es/docs/Web/HTTP/Headers>

Servidor HTTP en Node.js

- En Node.js existe un módulo nativo llamado “http”
- Tiene dos métodos fundamentales:
 - `http.createServer()`
 - `server.listen(port)`
- `createServer`, en su callback nos da dos parámetros, `req` y `res`.
- `req`, que representa la request:
 - `req.method`
 - `req.url`
 - `req.headers`
 - `req.body`

Servidor HTTP en Node.js

- `res (response)`
 - `res.writeHead(statusCode, headers)`
 - Por ejemplo:
 - `res.writeHead(200, {'Content-Type': 'application/json'})`
 - `res.write` -> envia un fragmento de respuesta.
 - `res.end` -> finaliza la respuesta
 - `res.setHeader`
 - Por ejemplo:
 - `res.setHeader('Content-Type', 'application/json')`

Servidor HTTP en Node.js

- El body de la request no viene en formato JSON, sino que viene en un stream.
- Hay que extraerlo de la siguiente forma:

```
let body = '';
req.on('data', chunk => body += chunk);
req.on('end', () => {
  // Hacer algo con el body
  const { name, description } = JSON.parse(body);
  res.writeHead(201, { 'Content-Type':
'application/json' });
  res.end(JSON.stringify(newGame));
});
```


Servidor HTTP en Node.js

- Una posible forma de estructurar el servidor:

Rutas

Controladores

Servicios

Modelos

Repositorios

Tests

etc

Servidor HTTP en Node.js

Ejercicio:

Los 3 grupos deberán escribir un servidor HTTP. Cada servidor deberá estar estructurado de la forma vista, brindando acceso a los recursos (solo a través de métodos GET y POST)

Grupo 1: servidor que maneja datos de libros.

Grupo 2: servidor que maneja datos de juegos.

Grupo 3: servidor que maneja datos de personajes históricos.