

Using pure
Node.js

NodeJs

Develop Web App

(HTTP Server,
HTTP GET, HTTP POST)



Fernando Saez

HTTP

HTTP (HyperText Transfer Procolol) o (Protocolo de transferencia de hipertexto).

Permite la transferencia de datos entre un cliente (Navegador) y un servidor web.

protocolo://host[:puerto][/ruta y archivo][?parámetros]

<https://www.google.com/search?q=perro>

Funcionamiento de HTTP

- El protocolo es iniciado por el cliente con un "request" que es casi siempre contestado por el servidor con un "response".
- Un request genérico tiene la forma:

METODO URI PROTOCOLO CrLf
HEADERS* CrLf
Datos

METODO: GET o POST

URI: http://unserver/un_recurso

PROTOCOLO: HTTP/1.1

CrLf: Retorno de carro + nueva línea.

Ej: GET <http://www.yahoo.com> HTTP/1.1

User-Agent: Mozilla/5.0 (Windows NT 5.1) Gecko/20060909

Firefox/1.5.0.7

Accept: text/html, image/png, */*

Accept-Language: en-us,en;q=0.5

Cookie: rememberme=true;

PREF=ID=21039ab4bbc49153:FF=4

Datos...

Funcionamiento de HTTP

- El server responde con un “response” de la forma:

PROTOCOLO STATUS VALOR CrLf

HEADERS*

Content-Type: TIPO CrLf

Datos

HTTP/1.1 200 OK

Date: Mon, 12 Jun 2000 14:04:28 GMT

Server: Apache/1.3.9 (Unix)

ApacheJServ/1.1

Connection: close

Content-Type: text/html

Datos...




Funcionamiento de HTTP

```
<HTML>
<HEAD>
<TITLE>Ejemplo</TITLE>
</HEAD>
<BODY>
Hola esta es una prueba
<IMG SRC="prueba.gif">
</BODY>
</HTML>
```

BROWSER	SERVER
GET http://www.prueba.com/index.html HTTP / 1.1	HTTP/1.1 200 OK Date: Tue, 13 Jun 2000 14:15:45 GMT Server: Apache/1.3.9 (Unix) PHP/4.0.0 Last-Modified: Tue, 13 Jun 2000 14:09:05 GMT ETag: "5804d-73-39464081" Accept-Ranges: bytes Content-Length: 115 Connection: close Content-Type: text/html <HTML> <HEAD> <TITLE>Ejemplo</TITLE> </HEAD> <BODY> Hola esta es una prueba </BODY> </HTML>
GET http://www.prueba.com/prueba.gif HTTP / 1.1	HTTP/1.1 200 OK Date: Tue, 13 Jun 2000 14:18:22 GMT Server: Apache/1.3.9 (Unix) PHP/4.0.0 Last-Modified: Tue, 13 Jun 2000 14:07:36 GMT ETag: "5804e-2b2-39464028" Accept-Ranges: bytes Content-Length: 690 Connection: close Content-Type: image/gif GIF89aGÖÿ11B99ZRJkcR-œk¼¼¼Æµ{µ,,{ZÖÆEi ÞœçÖ"-{5R5kJ,,{RkcBœ"k¥{Jµ (CORTADO)

Métodos (o verbos) de HTTP

- Los métodos HTTP se consideran **seguros** si no alteran el estado del servidor.
- **Idempotencia** significa que varias solicitudes idénticas tendrán el mismo resultado.

HTTP Method	Safe	Idempotent
GET		
POST		
PUT		
DELETE		
OPTIONS		
HEAD		

Status de Response

- El primer dígito del código de estado define la clase de Response. Los últimos 2 dígitos no tienen categorización.
 - 1xx: **Informacional** – Request recibido, proceso continua.
 - 2xx: **Exitoso** – La acción fue recibida exitosamente, entendida y aceptada
 - 3xx: **Redirección** – Debe realizar una acción, en orden a completar el request
 - 4xx: **Client Error** – El request contiene error en la sintaxis o no puede ser procesado.
 - 5xx: **Server Error** – El server fallo al procesar un request válido.

Status de Response

```
| "101" ; Section 10.1.2: Switching Protocols
| "200" ; Section 10.2.1: OK
| "201" ; Section 10.2.2: Created
| "202" ; Section 10.2.3: Accepted
| "203" ; Section 10.2.4: Non-Authoritative Information
| "204" ; Section 10.2.5: No Content
| "205" ; Section 10.2.6: Reset Content
| "206" ; Section 10.2.7: Partial Content
| "300" ; Section 10.3.1: Multiple Choices
| "301" ; Section 10.3.2: Moved Permanently
| "302" ; Section 10.3.3: Found
| "303" ; Section 10.3.4: See Other
| "304" ; Section 10.3.5: Not Modified
| "305" ; Section 10.3.6: Use Proxy
| "307" ; Section 10.3.8: Temporary Redirect
| "400" ; Section 10.4.1: Bad Request
| "401" ; Section 10.4.2: Unauthorized
| "402" ; Section 10.4.3: Payment Required
| "403" ; Section 10.4.4: Forbidden
| "404" ; Section 10.4.5: Not Found
| "405" ; Section 10.4.6: Method Not Allowed
| "406" ; Section 10.4.7: Not Acceptable

| "407" ; Section 10.4.8: Proxy Authentication Required
| "408" ; Section 10.4.9: Request Time-out
| "409" ; Section 10.4.10: Conflict
| "410" ; Section 10.4.11: Gone
| "411" ; Section 10.4.12: Length Required
| "412" ; Section 10.4.13: Precondition Failed
| "413" ; Section 10.4.14: Request Entity Too Large
| "414" ; Section 10.4.15: Request-URI Too Large
| "415" ; Section 10.4.16: Unsupported Media Type
| "416" ; Section 10.4.17: Requested range not satisfiable
| "417" ; Section 10.4.18: Expectation Failed
```


Developing a website with



Algunas aclaraciones!!

¿Como iniciar un Proyecto?

Requisitos: Tener Node instalado

1.- Crear un directorio que funcione como contenedor del Proyecto. > `mkdir appnode`

2.- > `npm init -y`

3.- Crear la aplicación: `Crear index.js` y editarlo.

4.- Ejecutar la aplicación > `node index.js`

Index.js – Servidor Web

```
var http = require('http');  
http.createServer(function (req, res) {  
    res.writeHead(200, {'Content-  
    Type': 'text/plain'});  
    res.write('Hello World!');  
    res.end();  
}).listen(8081);
```

requestListener



// Consola mostrara el siguiente mensaje

console.log('Server running at http://127.0.0.1:8081/');

Otro ejemplo

```
const http=require('http');
const servidor=http.createServer((req,res) => {
  res.writeHead(200, {'Content-Type': 'text/html'});
  res.write(` <!doctype html><html><head></head>
              <body><h1>Sitio en desarrollo
</h1></body></html>` );
  res.end();
});

servidor.listen(8888);
console.log('Servidor web iniciado');
```

un ejemplo mas

```
const http = require("http");
http
  .createServer((req, res) => {
    const { method, url, headers } = req;
    res.writeHead(200, { "content-type": "text/plain" });
    res.write(`Method: ${method} \n`);
    res.write(`Url: ${url} \n`);
    res.end(`Headers: ${JSON.stringify(headers)} `);
  })
  .listen(8080);
```

```
Method: GET
Url: /
Headers: {"host":"localhost:8080","connection":"keep-alive","cache-control":"max-age=0","sec-ch-ua":"\\"Not?A_Brand\\";v=\\"8\\", \\"Chromium\\";v=\\"108\\", \\"Google Chrome\\";v=\\"108\\", \\"sec-ch-ua-mobile\\";\\"?0\\", \\"sec-ch-ua-platform\\";\\"Windows\\"","upgrade-insecure-requests":"1","user-agent":"Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36","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=0.9","sec-fetch-site":"none","sec-fetch-mode":"navigate","sec-fetch-user":"?1","sec-fetch-dest":"document","accept-encoding":"gzip, deflate, br","accept-language":"es-ES,es;q=0.9"}
```

Routing

```
const http = require("http");
http
  .createServer((request, response) => {
    if (request.method === "GET" && request.url === "/home")
    {
      response.writeHead(200, { "content-type": "text/html" });
      response.write("<h3>Bienvenidos a mi sitio!!</h3>");
      response.end();
    } else {
      response.statusCode = 404;
      response.end();
    }
  })
  .listen(8080);
```

Otro ejemplo de routing

```
http.createServer(function (req, res) { // ....
  res.writeHead(200, {'Content-Type': 'text/html'});
  var url = req.url;
  if(url === '/about') {
    res.write(' Welcome to about us page');
    res.end();
  } else if(url === '/contact') {
    res.write(' Welcome to contact us page');
    res.end();
  } else {
    res.write('Hello World!');
    res.end();
  }
}
```

Requerir Módulos externos

```
const http = require("http");
const fsp = require("fs").promises;
//import fs from 'fs';
http
  .createServer((request, response) => {
    if (request.method === "GET" && request.url === "/cat") {
      fsp
        .readFile("./cat.gif")
        .then((imagen) => {
          response.writeHead(200, { "content-type": "image/gif" });
          response.write(imagen);
          response.end();
        })
        .catch((error) => {
          response.statusCode = 500;
          response.end();
        });
    } else {
      response.statusCode = 404;
      response.end();
    }
  })
  .listen(8080);
```


Factorizar...

Server

Routing

Handler
Request

App o
Index

....

```
http.createServer(function (req, res) { // ....
  res.writeHead(200, {'Content-Type': 'text/html'});
  var url = req.url;
  if(url === '/about') {
    res.write(' Welcome to about us page');
    res.end();
  } else if(url === '/contacto') {
    res.write(' Welcome to contact us page');
    res.end();
  } else {
    res.write('Hello World!');
    res.end();
  }
}
```

Factorizando (1)

Modularizamos en 3 archivos:

1. router.js
2. HandlerRequest.js
3. server.js
4. index.js

Preliminar

```
//router.js
function route() {
  if(url === '/about') {
    res.write(' Welcome to about us
page');
    res.end();
  } else if(url === '/contacto') {
    res.write(' Welcome to contact us
page');
    res.end();
  } else {
    res.write('Hello World!');
    res.end();
  }
}
exports.route = route;
```

Factorizando (1.1) Router.js

//router.js

```
function route(handle, request, response) {  
  let pathname = request.url;  
  if (typeof handle[pathname] === 'function') {
```

Definitivo!!

1

```
    handle[pathname](response, request);  
  } else {  
    console.log("No request handler found for " + pathname);  
    response.writeHead(404, {"Content-Type": "text/html"});  
    response.write("404 Not found");  
    response.end();  
  }  
}  
exports.route = route;
```

Factorizando (2) – requestHandler.js

//handlerRequest.js

```
function index(response, request) {  
  response.writeHead(200, {"Content-Type": "text/html"});  
  response.write('<h1> Hello World </h1>');  
  response.end();  
}
```

```
function about(response, request) {  
  response.writeHead(200, {"Content-Type": "text/html"});  
  response.write('<h1> Welcome to about us page </h1>');  
  response.end();  
}
```

...

```
exports.index = index;  
exports.about = about;  
exports.contacto = contacto;
```

Factorizando(3) – server.js

//server.js

```
var http = require("http");  
var url = require("url");
```

```
1 function start(route, handle) {  
    function requestListener(request, response) {  
        3 route(handle, request, response);  
        http.createServer(requestListener).listen(8888);  
        console.log("Servidor iniciado.");  
    }  
2 exports.start = start;
```

Factorizando (4) – index.js

//index.js

```
var server = require("./server");  
var router = require("./router");  
var requestHandlers = require("./requestHandlers");
```

```
var handle = {}
```

```
handle["/"] = requestHandlers.index;  
handle["/about"] = requestHandlers.about;  
handle["/contact"] = requestHandlers.contact;
```

```
server.start(router.route, handle);
```



**ALGUNA, PREGUNTA, DUDA,
SUGERENCIA, COMENTARIOS,
RECLAMOS,...!!???**