

Serveur Node



Hello World public de Node.js

```
const http = require('http');
const adresse = '127.0.0.1';
const port = 3000;

const server = http.createServer((requete, reponse) => {
  reponse.statusCode = 200;
  reponse.setHeader('Content-Type', 'text/plain');
  reponse.end('Hello World\n');
});

server.listen(port, adresse, () => {
  console.log(`Serveur qui roule à http://${adresse}:${port}/`);
});
```

Source: https://nodejs.org/en/about/
Exemple officiel

Tester avec le fureteur à http://localhost:3000/



Hello World de Service



Programme 'hello world' en tant que serveur

Fichier serveur-web.js

```
var http = require('http');
var repondeur = function(requete,reponse)
{
        console.log( 'execute response' );
}
var serveur = http.createServer(repondeur);
serveur.listen(3000);
```

- Lancer le service

C:\>cd nodejs

C:\nodejs>

nodejs **serveur-web**

_

- Autoriser le firewall
- Tester avec le fureteur à http://localhost:3000/

https://nodejs.org/api/https.html et https://nodejs.org/api/http.html

Programme 'hello world' en tant que serveur

Fichier allo-clients.js

```
var http = require('http');
var repondeur = function(requete,reponse){

        reponse.writeHead(200);
        reponse.end('allo visiteur');
}
var serveur = http.createServer(repondeur);
serveur.listen(3000);
```

- Lancer le service

C:\>cd nodejs

C:\nodejs>

nodejs **allo-clients**

_

- Autoriser le firewall
  ufw allow 3000/tcp
- Tester avec le fureteur à
  http://localhost:3000/

https://nodejs.org/api/https.html et https://nodejs.org/api/http.html

Programme 'hello world' en tant que serveur

Fichier allo-clients.js

```
var http = require('http');
var repondeur = function(requete,reponse){

        reponse.writeHead(200);
        reponse.write('allo visiteur');
        reponse.end();
}
var serveur = http.createServer(repondeur);
serveur.listen(3000);
```

- Lancer le service

C:\>cd nodejs

C:\nodejs>

nodejs **allo-clients**

_

- Autoriser le firewall
  ufw allow 3000/tcp
- Tester avec le fureteur à
  http://localhost:3000/

https://nodejs.org/api/https.html et https://nodejs.org/api/http.html



Anatomie d'une transaction



Étapes présentées dans ce guide





Composantes de la variable reponse

```
var requete = {
    headers: headers,
    method: method,
    url: url,
    body: body
  };

  reponse.write(JSON.stringify(reponse));
  reponse.end();

  // Note: the 2 lines above could be replaced with this next one:
  // reponse.end(JSON.stringify(reponse))
```



https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/



Lecture des url, method & headers

```
var method = requete.method;
var url = requete.url;

var entetes =requete.headers;
var userAgent = entetes['user-agent'];
```



https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/

Lecture des url, method & headers

```
const http = require('http');
var repondeur = function(requete,reponse)
{
/////////// ANALYSER la DEMANDE ////////
var methode = requete.method;
var url = requete.url;
console.log("methode = " + methode);
console.log("url = " + url);
/////////// PREPARE REPONSE //////
reponse.statusCode = 200;
reponse.setHeader('Content-type','text/plain');
// inserer un reponse.write() ici
reponse.end();
}
var serveur = http.createServer(repondeur);
serveur.listen(3000);
```



Préparation de la réponse

```
reponse.statusCode = 404;


reponse.setHeader('Content-Type', 'application/json');
reponse.setHeader('X-Powered-By', 'bacon');
```



https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/



Répondre à des url



Fonction Callback

Tester l'url

http://localhost:8080/mouton/liste
https://github.com/nadineprofesseur/service.bergerie.node/blob/e1def2e8bf28a2ed425321d221b784464ded7274/service.js

http://localhost:8080/mouton/3
https://github.com/nadineprofesseur/service.bergerie.node/blob/589a8bff47400d48a5749250e49499ddc0938f1d/service.js

Détection du 3 dans http://localhost:8080/mouton/3
https://github.com/nadineprofesseur/service.bergerie.node/blob/7b1ee457ba0f141ee0f35af8a87456e022ec17c7/service.js

Tester l'url

```
var moutonDAO = require('./MoutonDAO');
var http = require('http');

var repondre = async function(requete, reponse)
{
        if('GET' === requete.method)
        {
                if('/mouton/liste' === requete.url || '/mouton/liste/' === requete.url)
                {
                        var listeMoutons = await moutonDAO.listerMoutons();
                        reponse.end(JSON.stringify(listeMoutons));
                }
        }
        if('POST' === requete.method)
        {
        }
        reponse.end('');
}

var serveur = http.createServer(repondre);
serveur.listen(8080);
```

http://localhost:8080/mouton/liste

Url paramétrée

http://localhost:8080/mouton/3

```
var moutonDAO = require('./MoutonDAO');
var http = require('http');

var repondre = async function(requete, reponse)
{
	if('GET' === requete.method)
	{
		if('/mouton/liste' === requete.url || '/mouton/liste/' === requete.url)
		{
			var listeMoutons = await moutonDAO.listerMoutons();
			reponse.end(JSON.stringify(listeMoutons));
		}
		if(requete.url.match(/\/mouton\/[0-9]+\/?/))
		{
			reponse.end('match un seul mouton');
		}
	}
	if('POST' === requete.method)
	{
	}
```

Isoler le paramètre d'url

```
var moutonDAO = require('./MoutonDAO');
var http = require('http');

var repondre = async function(requete, reponse)
{
        if('GET' === requete.method)
        {
                if('/mouton/liste' === requete.url || '/mouton/liste/' === requete.url)
                {
                        var listeMoutons = await moutonDAO.listerMoutons();
                        reponse.end(JSON.stringify(listeMoutons));
                }
                if(trouvailles = requete.url.match(/\/mouton\/([0-9]+)\/?/))
                {
                        numero = trouvailles[1];
                        reponse.end('match un seul mouton ' + numero);
                        var mouton = await moutonDAO.demanderMouton(numero);
                }
        }
```

http://localhost:8080/mouton/3



Envoyer des données



Isoler le paramètre d'url

```
var moutonDAO = require('./MoutonDAO');
var http = require('http');

var repondre = async function(requete, reponse)
{
        if('POST' === requete.method)
        {
                console.log(requete.body.nomDuChamps);
                if('/mouton/ajouter' === requete.url || '/mouton/ajouter/' === requete.url)
                {
                        var succes = await moutonDAO.ajouterMouton(mouton);
                        reponse.end(JSON.stringify(succes));
                }
                if('/troupeau/ajouter' === requete.url || '/troupeau/ajouter/' === requete.url)
                {
                }
        }

A TESTER
```

http://localhost:8080/mouton/ajouter

http://localhost:8080/troupeau/ajouter

Isoler le paramètre d'url

```
var moutonDAO = require('./MoutonDAO');
var http = require('http');

var repondre = async function(requete, reponse)
{
        if('POST' === requete.method)
        {
                var data = '';
                requete.on('data', function(message) {data += message;});
                requete.on('end', function() {console.log('POST data: %s', data);});
                var formulaire = JSON.parse(data); var  mouton = new Mouton(formulaire);

                if('/mouton/ajouter' === requete.url || '/mouton/ajouter/' === requete.url)
                {
                        var succes = await moutonDAO.ajouterMouton(mouton);
                        reponse.end(JSON.stringify(succes));
                }
                if('/troupeau/ajouter' === requete.url || '/troupeau/ajouter/' === requete.url)
                {
                }
        }
```

http://localhost:8080/mouton/ajouter

http://localhost:8080/troupeau/ajouter



Exemples particuliers



Serveur echo

```
var http = require('http');

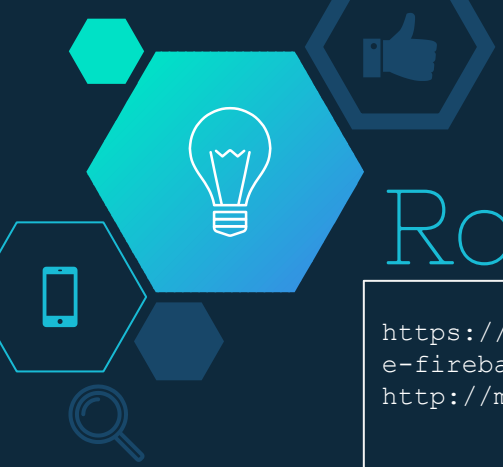
http.createServer(function(requete, reponse) {
  requete.on('error', function(err) {
    console.error(err);
    reponse.statusCode = 400;
    reponse.end();
  });
  reponse.on('error', function(err) {
    console.error(err);
  });
  if (requete.method === 'GET' && requete.url === '/echo') {
    request.pipe(response);
  } else {
    reponse.statusCode = 404;
    reponse.end();
  }
}).listen(8080);
```

https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/

Entrée du contenu du body

```
var body = [];
request.on('data', function(chunk)
{
  body.push(chunk);
})
.on('end', function()
{
  body = Buffer.concat(body).toString();
});
```

https://nodejs.org/en/docs/guides/anatomy-of-an-http-transaction/



Router Endpoint (avancé)

https://medium.freecodecamp.org/how-to-write-beautiful-node-js-apis-using-async-await-and-the-firebase-database-befdf3a5ffee
http://mherman.org/blog/2015/02/12/postgresql-and-nodejs/#.WgBVKmjWw2w





Node.js

Notes de cours
par Nadine
Giasson
St-Amand

