NODE JS

Mr Herifaniry Rakotoniaina

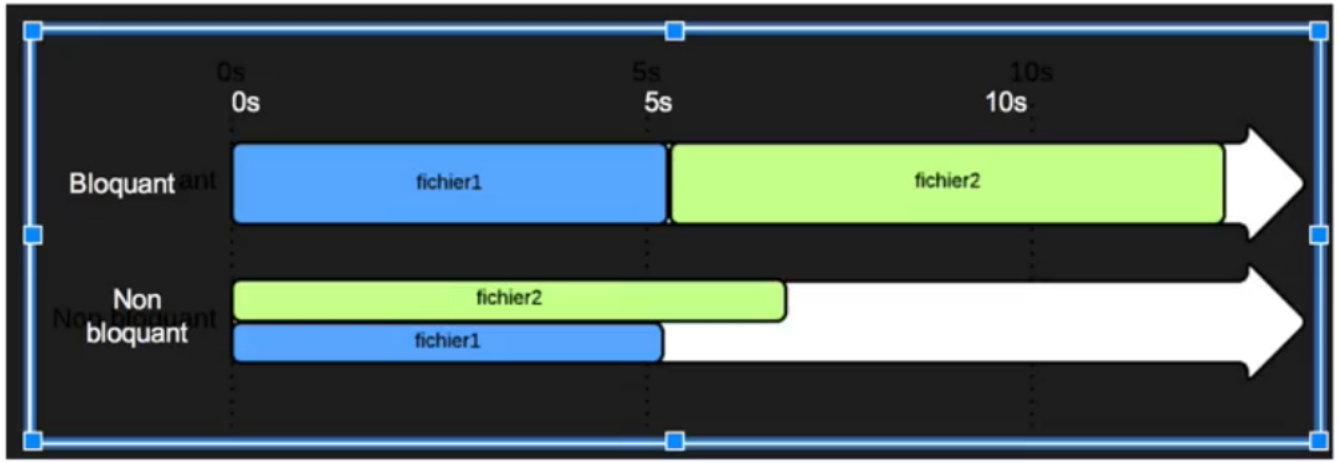
MCCI

Présentation

- **Node.js** est une plateforme logicielle libre et événementielle en Javascript, basé sur le moteur V8 Javascript Engine de Google utilisé notamment par le navigateur Chromium et Google Chrome.
- **Node.js** est un serveur utilisant comme language principal le Javascript.
- **Node.js** peut:
  - servir des pages HTML classiques
  - exécuter des codes en temps réel

Performance

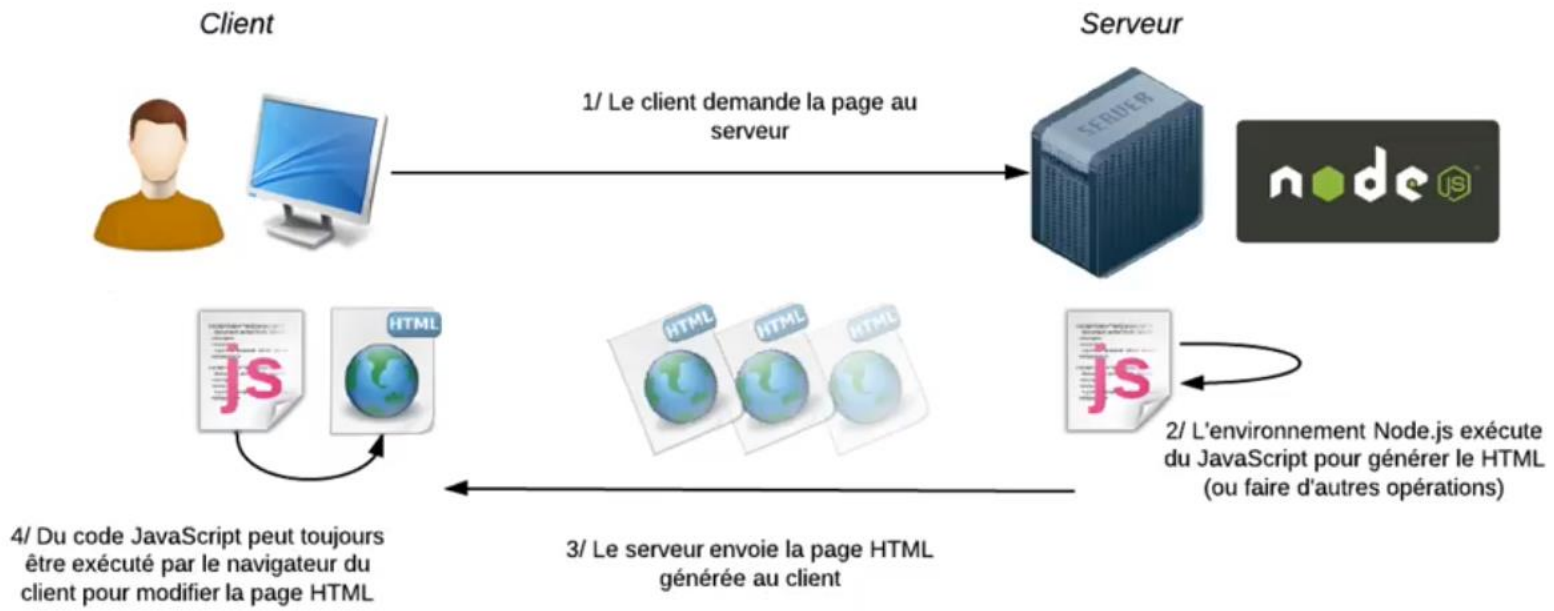
- Modèle événementiel, E/S non bloquantes, Node.js peut répondre à un nombre de requêtes infiniment plus conséquent qu'une même application développée avec une application LAMP classique.



Evolution de Javascript

- 3 évolutions :
  1. L'ère du **DHTML** durant les années 90 avec l'utilisation de navigateurs de type **Netscape** ou **Internet Explorer 5.5**
  2. L'ère de la **manipulation du DOM** avec des librairies types **JQuery** ou **Mootools** durant les années 2000
  3. L'ère **technologique** depuis les années 2010 avec des outils type **Node.js**

Javascript coté serveur



Cas d’utilisation

- Installation
  - https://nodejs.org, télécharger la version LTS (Long-term Support)
  - 12 mois + 18 mois maintenance

Exemple

- Hello world
  - Créez un fichier helloworld.js

```
console.log("Hello World");
```

  - Dans le terminal, lancer avec la commande

```
node helloworld.js
```

Objectifs

- Application accessible depuis un navigateur.
- L'utilisateur doit voir, depuis http::://domain/start, une page d'accueil affichant un formulaire de transfert de fichier.
- En soumettant le formulaire après avoir choisi une image, celle-ci doit être uploadée à l'adresse http://domaine/upload qui l'affichera une fois le transfert terminé.

Objectifs Code

- Créer un serveur HTTP
- Routeur URL
- Gestionnaires des requêtes
- Gestionnaire des données (POST, …)
- Modèle de vue pour renvoyer les éléments au navigateur
- Gestionnaire upload

CommonJs

- Il manque beaucoup de choses au standard Javascript pour devenir un langage capable de vivre en dehors du browser: Entrées/sortie, modules, appels systèmes, gestion de dépendances, lignes de commandes, etc.
- commonJS est un projet qui répond à ces besoins.
- NodeJS implémente l'API CommonJS 1.0 (annuaires des modules: npmjs.org)
- Chargement d'un module CJS identifié

Modules standards les plus utiles (1)

- **Process**: permet de récupérer les arguments de la ligne de commande, entre autres…
- **http**: pour la gestion du protocole HTTP
- **fs**: Système de fichier
- **url**: Gestion des URL
- **querystring**: gestion des paramètres HTTP

Modules standards les plus utiles (2)

- **Console** : pour les traces
- **Net** : pour créer des serveurs de sockets
- **Cluster** : répartition CPU multi cœurs
- **Os** : Utilitaire pour le système d’exploitation
- **Events** : Gestion des événements

Serveur HTTP (1)

- Répartir code en plusieurs fichiers:
  - Ex: serveur.js et index.js
- Créer le fichier server.js

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

http.createServer(function(request, response) {
  response.writeHead(200, {"Content-Type": "text/plain"});
  response.write("Hello World");
  response.end();
}).listen(8888);
```

- Lancer avec node server.js et tester sur : http://localhost:8888

Serveur HTTP(2)

- Analyse du code
  - Fonction auto execute **listen**

```
http.createServer(function(request, response) {
  response.writeHead(200, {"Content-Type": "text/plain"});
  response.write("Hello World");
  response.end();
}).listen(8888);
```

  - Fonction comme paramètre de **createServer()**

```
http.createServer(function(request, response) {
  response.writeHead(200, {"Content-Type": "text/plain"});
  response.write("Hello World");
  response.end();
}).listen(8888);
```

Server HTTP (2)

- Fonction comme paramètre dans Javascript

```
function say(word) {
  console.log(word);
}

function execute(someFunction, value) {
  someFunction(value);
}

execute(say, "Hello");
```

```
function execute(someFunction, value) {
  someFunction(value);
}

execute(function(word){ console.log(word) }, "Hello");
```

Server HTTP (3)

- Réécriture du code

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

function onRequest(request, response) {
  console.log("Requête reçue.");
  response.writeHead(200, {"Content-Type": "text/plain"});
  response.write("Hello World");
  response.end();
}

http.createServer(onRequest).listen(8888);
console.log("Démarrage du serveur.");
```

- Tester

Module server

- Exporter les fonctions
- Changer contenu server.js

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

function start() {
  function onRequest(request, response) {
    console.log("Request received.");
    response.writeHead(200, {"Content-Type": "text/plain"});
    response.write("Hello World");
    response.end();
  }
  http.createServer(onRequest).listen(8888);
  console.log("Démarrage du serveur.");
}

exports.start = start;
```

Fichier index

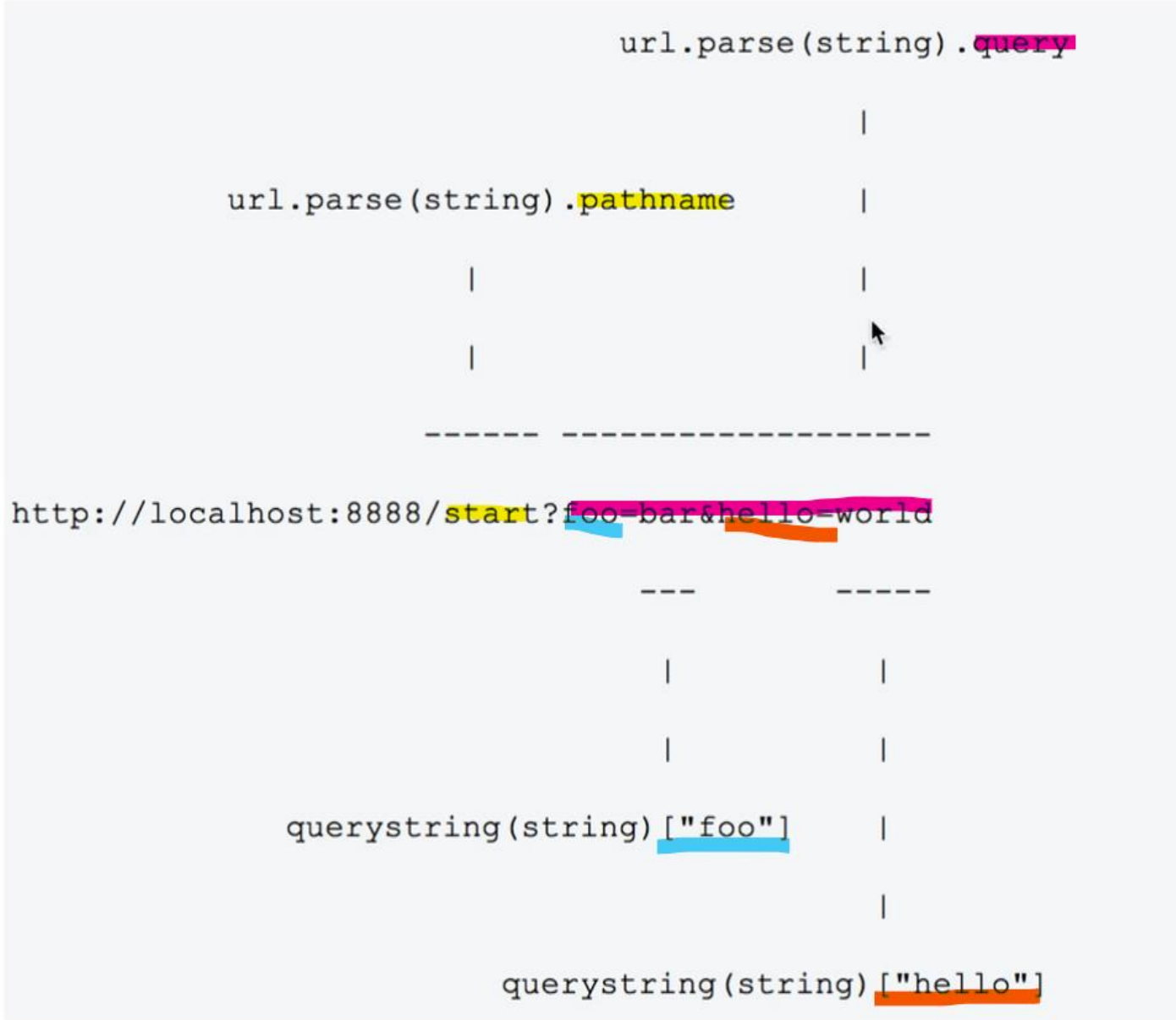
- Contenu index.js

```
var server = require("./server");
server.start();
```

Routage requête

- Pour rediriger les requêtes vers la bonne fonction de traitement
- Utilisation de l’objet **request**
- Utilisation de modules
  - url
  - querystring


url.parse(string).query
url.parse(string).pathname
http://localhost:8888/start?foo=bar&hello=world
querystring(string)["foo"]
querystring(string)["hello"]


url.parse(string).query
url.parse(string).pathname
http://localhost:8888/start?foo=bar&hello=world
querystring(string)["foo"]
querystring(string)["hello"]

Code server.js

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

function start() {
  function onRequest(request, response) {
    var pathname = url.parse(request.url).pathname;
    console.log("Requête reçue pour le chemin " + pathname + ".");
    response.writeHead(200, {"Content-Type": "text/plain"});
    response.write("Hello World");
    response.end();
  }
  http.createServer(onRequest).listen(8888);
  console.log("Démarrage du serveur.");
}
```

Code router.js

- Code router.js

```
function route(pathname) {
  console.log("Début du traitement de l'URL " + pathname + ".");
}

exports.route = route;
```

Inversion de Dépendance

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

function start(route) {
  function onRequest(request, response) {
    var pathname = url.parse(request.url).pathname;
    console.log("Requête reçue pour le chemin " + pathname + ".");
    route(pathname);
    response.writeHead(200, {"Content-Type": "text/plain"});
    response.write("Hello World");
    response.end();
  }
  http.createServer(onRequest).listen(8888);
  console.log("Démarrage du serveur.");
}

exports.start = start;
```

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

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

Router vers gestionnaire(1)

- requestHandlers.js

```
function start() {
  console.log("Le gestionnaire 'start' est appelé.");
}

function upload() {
  console.log("Le gestionnaire 'upload' est appelé.");
}

exports.start = start;
exports.upload = upload;
```