Node.js

JavaScript côté serveur

plb consultant

Bienvenue

- Qui êtes-vous ?
- Quel poste occupez-vous ?
- Quelles sont vos connaissances en javascript ?
- Quelles sont vos connaissances en node.js ?
- Quelles sont vos attentes ?

Adrien Pignon

- Développeur Web depuis 6 ans
- Formateur depuis 2 ans
- Spécialité : Javascript, Typescript, PHP

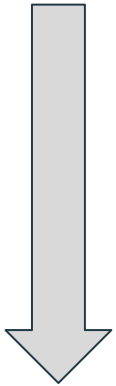
Sujet et Objectifs

À l'issue de cette formation **Node.js**, vous aurez acquis les connaissances et compétences nécessaires pour :

- Connaître le fonctionnement et les usages courants de la plateforme Node.js
- Connaître les spécificités de la programmation côté serveur en JavaScript
- Savoir installer et configurer un serveur Node.js
- Connaître les particularités de la programmation asynchrone et celles de la programmation orientée événements
- Savoir paramétrer et manipuler l'API Node.js et son serveur d'API REST
- Savoir se connecter à une base de données
- Mettre en œuvre les bonnes pratiques Node.js

Historique

Node.js est une plateforme logicielle libre en JavaScript



- **1995** : Création de **Javascript**
- **2009 :** Création de **Node.js** par **Ryan Dahl**
- **2018 :** plus de **20 millions** de sites au total sur Internet utilisent **Node.js.**

Historique

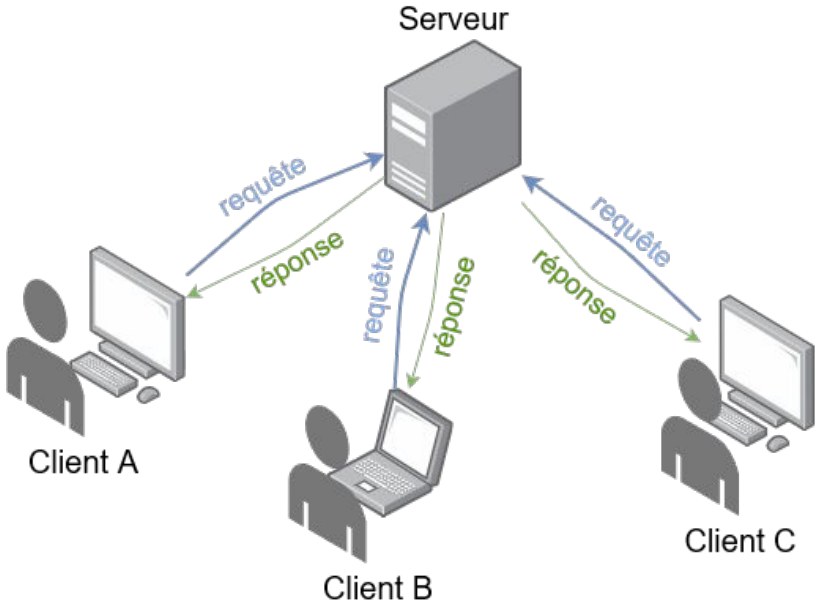
Qui utilise node.js :

- **Twitter**
- **Spotify**
- **Uber**
- **Netflix**
- **eBay**
- **Reddit**
- **LinkedIn**
- **Trello**

Comment cela fonctionne (techniquement) ?

- **V8** est **un moteur JavaScript** open-source développé par le projet Chromium pour les navigateurs Web Google Chrome et Chromium
- **V8** compile directement le code **JavaScript** en **code machine natif** avant de l'exécuter.
- **Node.js** est **un environnement bas niveau permettant l'exécution de JavaScript côté serveur.**

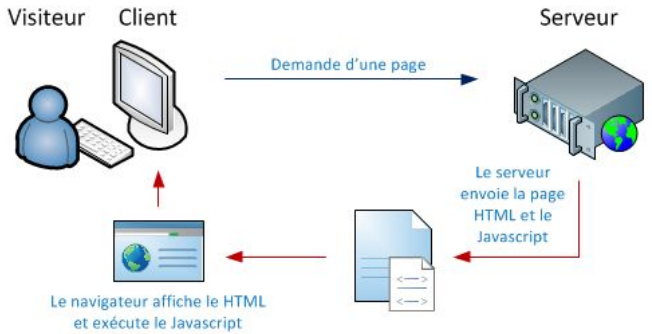
Comment cela fonctionne (techniquement) ?



Comment cela fonctionne (techniquement) ?

Les langages "client-side" s'exécutent dans le navigateur (le client) : HTML, CSS, Javascript

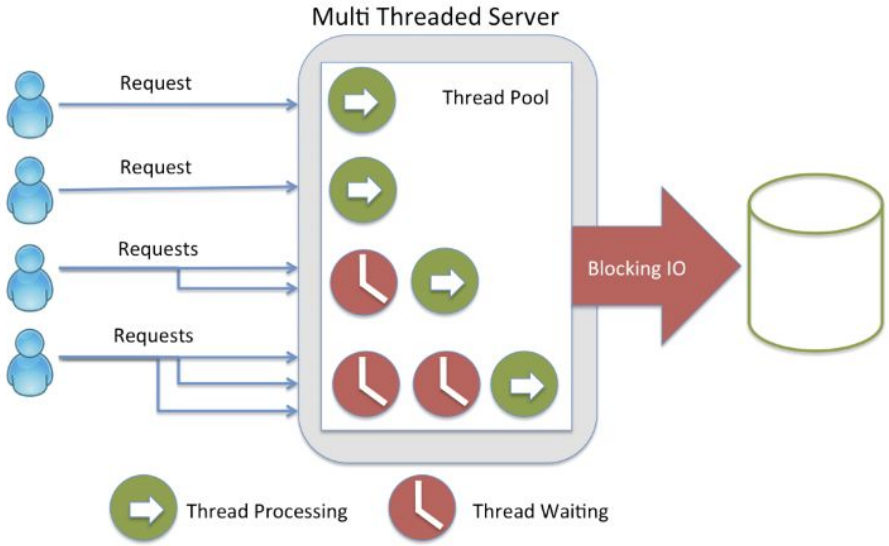
Les langages "server-side" s'exécutent dans le serveur : PHP, Python, Ruby, Javascript…



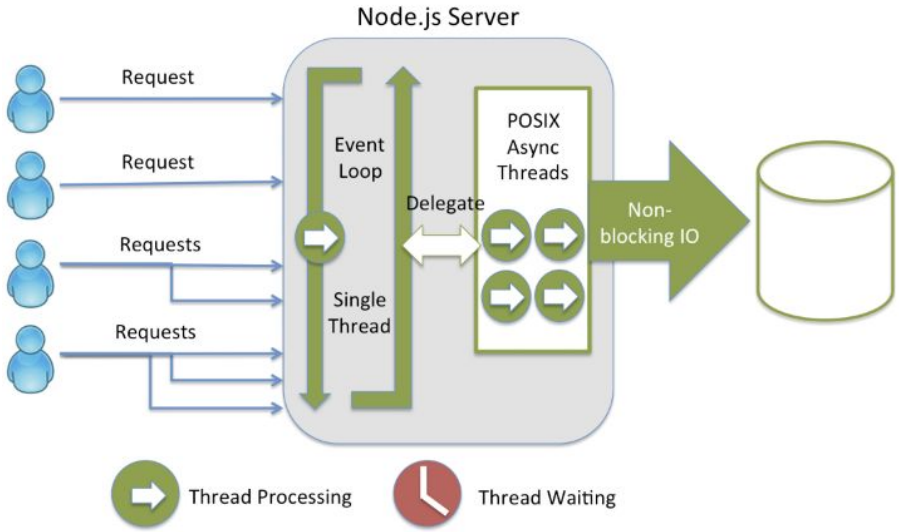
Pourquoi utiliser la programmation événementielle

- La programmation événementielle utilise les événements pour déterminer l'exécution d'un programme.
- Un événement peut être une action de l'utilisateur.
- Cela permet de structurer son code de manière indépendante en fonction des actions de l’utilisateur.
- Le code ne s'exécute qu’en fonction d’un événement précis.

Approche non-bloquante d'une application : avantages et limites



Approche non-bloquante d'une application : avantages et limites



Approche non-bloquante d'une application : avantages et limites

Avantage :

- Pour les application hautement concurrentielle (requêtes simultanées)
- Moins d’attente

Limites :

- Imbrication du code
- Single thread => Ne rien faire qui demande beaucoup de calculs
- Bas niveau => Créer le serveur web au niveau de notre code

Plan du cours

1. Installation du serveur node.js
2. Le gestionnaire d'extensions NPM
3. Premier cas concret
4. Un serveur web en quelques lignes
5. Utilisation de Node.js en REPL
6. L'injection de fonctions en JavaScript
7. Travaux pratiques

Installation du serveur node.js

- Téléchargemnt : https://nodejs.org/
- Version : LTS (18)

Le gestionnaire d'extensions NPM

- Node.js Package Manager (npm) est le gestionnaire de paquets par défaut et le plus populaire dans l'écosystème Node.js.

- Il est principalement utilisé pour installer et gérer des modules externes dans un projet Node.js.

Le gestionnaire d'extensions NPM

- **Les dépendances** sont des éléments dont la présence est obligatoire au bon fonctionnement d'une application.
- Ce sont des bibliothèques (jQuery, Bootstrap pour un site Web) ou des frameworks (Angular, Express.js) voire d'autres applications (compilateur, transpileur).
- **Les gestionnaires de paquets** automatisent le téléchargement, l'installation des dépendances ainsi que la gestion des versions de celle-ci.
- Le terme **paquet** désigne un conteneur de fichiers informatiques avec les procédures d'installation.
- **npm** est le gestionnaire de dépendances le plus connu, il est fourni avec Node.js

Le gestionnaire d'extensions NPM

- Créer un projet node : **npm init**

- Structure du projet stocké dans **package.json**

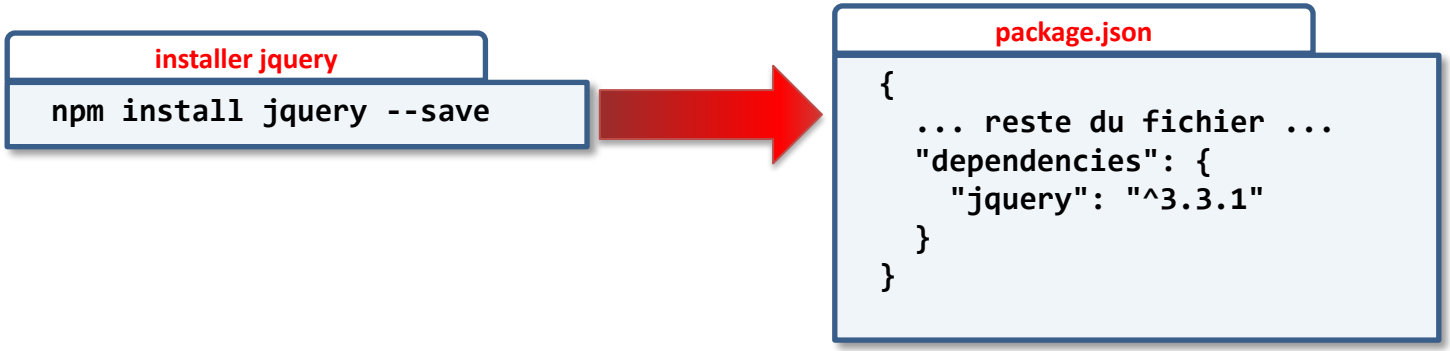
Le gestionnaire d'extensions NPM

package.json

```
{
    "name": "tp1",
    "version": "1.0.0",
    "description": "",
    "main": "index.js",
    "scripts": {
        "test": "echo \"Error: no test specified\" && exit 1"
    },
    "author": "",
    "license": "ISC"
}
```

Le gestionnaire d'extensions NPM

La commande pour installer une dépendance : **npm install uneDépendance --save**



Le fichier **package.json** se voit ajouter le nom et **la version de la dépendance**.

La commande a créée le répertoire **node_modules** qui contiendra **les dépendances.**

Premier cas concret

Afficher un “Hello world !” dans votre console :

1. Créer un fichier script.js
2. Ajouter console.log(‘Hello world !’); à l’intérieur
3. Exécuter votre script avec la commande node script.js

Un serveur web en quelques lignes

```
const http = require('node:http');

const hostname = '127.0.0.1';
const port = 3000;

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

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Utilisation de Node.js en REPL

La commande “node index.js” exécute le fichier js

On peut aussi utiliser et personnaliser repl dans notre environnement node :

```
const repl = require('repl');
repl.start();
```

Récupérer les arguments du terminal :

```
console.log('arg 1 :', process.argv[2]);
console.log('arg 2 :', process.argv[3]);
```

```
$ node process.js test yes
arg 1 : test
arg 2 : yes
```

Travaux pratiques

- Créer un projet avec npm
- Créer une commande qui permet de lancer le serveur

Exercice 1

Phase 1:

- Créez un autre app.js fichier, appeler calculator.js Lorsque nous appelons node app.js, nous devrions afficher dans la console ce qui suit : The sum of 3 & 7 is: 10 The multiplication of 3 & 7 is: 21
- app.js ne devrait afficher dans la console que le résultat de l'appel des fonctions sum et multiplication
- Ces méthodes doivent être définies (et exportées) dans calculator.js

Phase 2

- Créez un dossier appelé opérations et créez dans chaque dossier un fichier exportant chaque opération nécessaire dans le app.jsfichier principal
- app.js operations/ sum.js multiplication.js substraction.js division.js

Phase 3

- Installez le module “moment” et utilisez-le pour afficher l'heure actuelle de cette manière :Today is : Monday, October 31st 2016, 10:08:34 am The sum of 3 & 7 is: 10 The multiplication of 3 & 7 is: 21

Exercice 2

- Écrivez un programme node.JS qui exécute une connexion au serveur et renvoie un message de réussite comme "Succès, j'écoute depuis le port : 3000"

Astuce : Vous avez besoin du module npm - http

- En plus : Définir le port de la connexion à l'aide d'une variable d'environnement PORT

Exercice 3

- Afficher une page web :