



Développement Backend – Node JS

INTERVENANT : THOMAS JAMME

PARTIE 1 : ENVIRONNEMENT
ET INSTALLATION

Environnement et installation

- Javascript
- Node.js
- Configuration de l'environnement de travail
- Le gestionnaire de dépendance NPM
- Initialisation d'un projet



Javascript

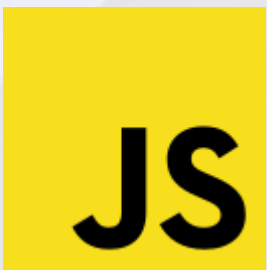


JS



JavaScript - Introduction

- JavaScript n'a d'abord AUCUN rapport avec le langage JAVA. Il s'agit simplement d'une erreur de marketing.
- Première version sortie en 1996.
- Langage de script léger interprété et orienté objet.
- Principalement connu comme le langage de script des applis web.
- JavaScript se base sur la norme ECMAScript 6 (ES6).
- Il a permis l'émergence d'application web monopage grâce à Ajax et les Framework tels que Angular.js, React ou encore Vue.js.
- Ces Framework sont incontournables dans le monde du web et de plus en plus présent dans le développement mobile.



JavaScript – Déclaration

Déclaration de variable

- Pour déclarer une variable, il n'est pas nécessaire d'utiliser de type. Javascript est un **langage non typé**.
- En revanche, depuis ES6, il faut préciser quelle est la vocation de la variable : **doit-elle être modifiée ou non ?**
- Si la variable doit être modifiée, il faut préfixer la déclaration de variable par « **let** » ou par « **const** » si elle ne doit pas l'être.
- La question du point virgule : le débat fait rage, mais pour faire simple : faites ce qui vous va le mieux !

```
let maVariableModifiable = "toto"  
const maVariableFixe = "tutu"
```

JavaScript – Les objets standard

Objet	Syntaxe
Chaine de caractères	<code>let str = "toto"</code>
Entier	<code>let int = 1</code>
Flotant	<code>let float = 1,2</code>
Tableau	<code>let tab = []</code>
Objet	<code>let obj = {}</code>

JavaScript – Les objets standard

Objet	Syntaxe
Fonction	<pre>function maFonction(param1, param2) { ...code... }</pre>
Fonction format ES6	<pre>const maFonction = (param1, param2) => { ...code... }</pre>

JavaScript – Expressions et opérateurs

Les opérateurs arithmétiques

- Addition : +
- Soustraction: -
- Division : /
- Multiplication: *
- Puissance: **

Les opérateurs de comparaison

- Plus grand que : >
- Plus petit que : <
- Plus grand ou égal que : >=
- Plus petit ou égal que : <=
- Egalité faible: ==
- Egalité forte : ===
- Inégalité faible : !=
- Inégalité forte: !==

JavaScript – Expressions et opérateurs

Les opérateurs logiques

- ET : &&
- OU: ||

Opérateur conditionnel ternaire

- (condition: ? siVrai : siFaux)
- *Ex : $a > b ? isValid = true : isValid = false$*

Les affectation

- Affecter une valeur à une variable : =

JavaScript – Instructions

Instructions conditionnelles

Instruction	SI	SINON SI	SINON
Syntaxe	<pre>if (condition) { ...code... }</pre>	<pre>if (condition) { ...code... } else if (condition2) { ...code... }</pre>	<pre>if (condition) { ...code... } else { ...code... }</pre>

JavaScript – Instructions

Les boucles

Instruction	TANT QUE	POUR	POUR CHAQUE
Syntaxe	<pre>while (condition) { ...code... }</pre>	<pre>for (let i = 0; i < limite; i++) { ...code... }</pre>	<pre>array.forEach(element => { ...code... });</pre>

JavaScript – Instructions

Instructions d'exécution

Instruction	ARRETER	CONTINUER	RETOURNER
Syntaxe	break	continue	return
	Arrête la boucle et en sort.	Arrête l'itération courante et passe à la suivante.	Termine la fonction et retourne une valeur.

JavaScript – Instructions

Autre instruction de déclaration

Instruction	CLASSE
Syntaxe	<pre>class maClasse { constructor(v1, v2) { this.param = v1 * v2 } }</pre>
	Permet de créer une classe avec un constructeur. This permet au éléments de la classe courante.

JavaScript – Instructions

Instruction d'import / exports

Instruction	IMPORTER	EXPORTER
Syntaxe	import exportParDefaut from "nom-module"	export data
	Importe un module.	Exporte data. Data peut être une fonction, une classe, etc...

JavaScript – Exercice

Exercice : Créer une application capable de calculer un volume

- Allez sur le GitLab pour télécharger la base de l'exercice : <https://gitlab.com/ThomasJamme/esn-node-js>
- Formule du volume : largeur x longueur x hauteur
- Affichez une erreur si un des paramètre est nul ou vide.

Node.js



Node JS - Introduction

- Node JS est une plateforme logicielle libre écrite en JavaScript.
- Première version sortie en 2010
- Il permet le développement d'un serveur HTTP
- Javascript est utilisé **côté serveur**
- Permet de mettre en place une **API**
- De nombreux Frameworks ont été créés tels que **Express** permettant de développer une API ou Ionic dédié au développement mobile.
- Node JS est asynchrone
- De nombreuses sociétés utilisent Node.js : Groupon, LinkedIn, PayPal, Slack...



GROUPON[®]



 Microsoft

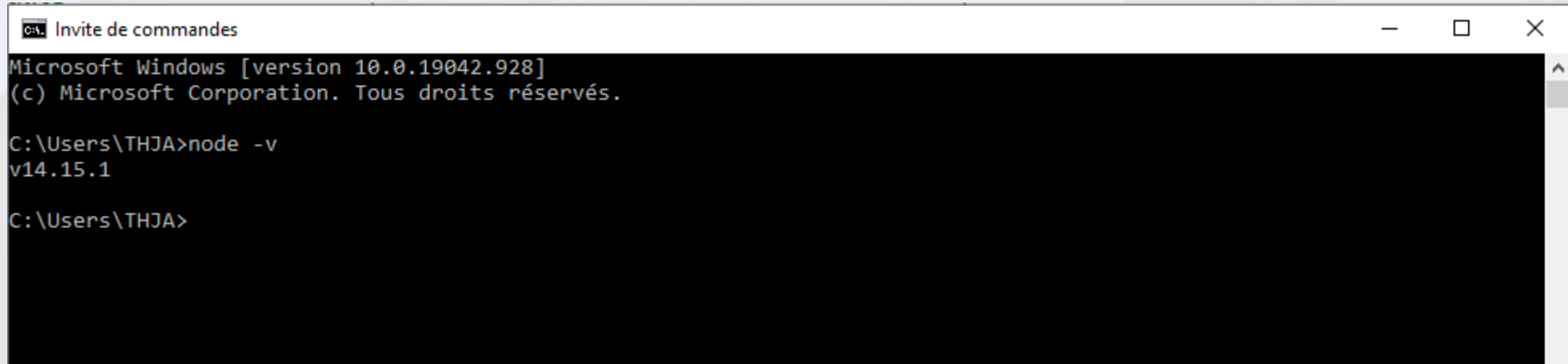


Configuration de l'environnement de travail



Environnement de travail - Installation de Node JS

- Pour installer Node.js, rendez-vous sur le site officiel de Node.js : <https://nodejs.org/>
- Télécharger la dernière version recommandée pour la plupart des utilisateurs
- Procédez à l'installation
- Pour être sûr que Node.js est bien installé, ouvrez un invite de commande et tapez « **node -v** »
- Le résultat attendu doit présenter la version de **Node.js** installée sur votre ordinateur.



```
Invite de commandes
Microsoft Windows [version 10.0.19042.928]
(c) Microsoft Corporation. Tous droits réservés.

C:\Users\THJA>node -v
v14.15.1

C:\Users\THJA>
```

Environnement de travail - Installation Visual Studio Code

- L'IDE proposé pour ce cours est Visual Studio Code
- Vous pouvez utiliser un autre IDE si vous le souhaitez
- Pour installer Visual Studio Code, rendez vous sur le site officiel <https://code.visualstudio.com/>
- Téléchargez la version pour Windows
- Installez Visual Studio Code
- Cet IDE est très léger, il peut être lancé en quelques secondes.



Visual Studio Code

Environnement de travail – Installation de PostMan

- PostMan est un excellent outil permettant de tester les **APIs**
- Téléchargez PostMan sur le site officiel :
<https://www.postman.com/downloads/>
- Installez l'application



POSTMAN

NPM



NPM

- **Npm** (Node Package Management) est le **gestionnaire de paquets officiel** de Node.js
- Npm est **automatiquement installé** avec Node.js
- Le site de npm (npmjs.com), regroupe l'ensemble des librairies disponibles pour Node.js
- Ce gestionnaire de paquets va permettre de gérer, entres autres, les actions suivantes :
 - **Installer des packages** et librairies utiles à votre code
 - **Initialiser un projet**
 - **Exécuter des commandes spécifiques** décrites dans le package.json
- Dans chaque projet en Node.js utilisant npm, on trouvera un fichier nommé **package.json**
- Le fichier **package.json** contient la liste des librairies devant être installées pour faire tourner le serveur.
- Pour installer ces librairies, il faudra exécuter la commande **npm install**
- Un dossier nommé « **node_module** » sera alors créé automatiquement et contiendra les librairies installées.



NPM

- Si vous souhaitez ajouter une librairie en particulier, il suffira d'exécuter la commande suivante : **npm install <nom_du_package>**
- Ex: **npm install mongoose** pour installer le package mongoose utile pour se connecter à une base de donnée MongoDB
- Le nom de la librairie sera alors automatiquement ajouté dans la section dependencies du fichier **package.json**

```
{
  "name": "chienmixbackend",
  "version": "1.0.0",
  "description": "Backend de l'appli chien mix",
  "main": "server.js",
  "scripts": {
    "test": "echo \\\"Error: no test specified\\\" && exit 1",
    "start": "node server.js"
  },
  "author": "Thomas Jamme",
  "license": "ISC",
  "dependencies": {
    "body-parser": "^1.19.0",
    "dotenv": "^8.2.0",
    "express": "^4.17.1",
    "fs": "0.0.1-security",
    "https": "^1.0.0",
    "mongoose": "^5.12.1"
  }
}
```

Initialisation d'un projet



Node JS – Initialisation d'un projet

- Pour initialiser un projet, créez un répertoire du nom de votre choix placez-vous dedans.
- Ouvrez-y un invite de commande
- Tapez **npm init**

```
$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help init` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (dev) esn-node|
```

- Renseignez les informations demandées:
 - **package- name** : nom du projet
 - **Version** : 1.0.0 par défaut
 - **Description** : Décrivez rapidement le projet
 - **Entry point**: Gardez index.js
 - **Test command, git repository, keyword** : vide
 - **Author**: votre adresse mail
 - **Licence**: ISC par défaut

```
package name: (dev) esn-node
version: (1.0.0)
description: Serveur backend en node.js
entry point: (index.js)
test command:
git repository:
keywords:
author: thomas.jamme@beproject.fr
license: (ISC)
```

Node JS – Initialisation d'un projet

- Validez les informations saisies
- Le dossier contient alors un seul fichier nommé **package.json**
- Ouvrez alors le dossier avec Visual Studio Code
 - File > Open Folder
 - Choisissez le dossier de votre projet
- Créez un fichier nommé index.js
- Placez le contenu suivant dedans :

```
const http = require('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');
});

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



```

1 // Import de la librairie http
2 const http = require('http');
3
4 // Définition de l'adresse IP du serveur. Il s'agit d'un serveur local.
5 const hostname = '127.0.0.1';
6
7 // Définition du port sur lequel doit tourner le serveur
8 const port = 3000;
9
10 // Création du serveur
11 // Il renverra toujours une réponse avec le code HTTP 200 lorsqu'il sera appelé
12 // La réponse contiendra le texte "Hello World"
13 const server = http.createServer((req, res) => {
14   res.statusCode = 200;
15   res.setHeader('Content-Type', 'text/plain');
16   res.end('Hello World');
17 });
18
19 // Mise en écoute du serveur
20 // Affichage d'un log indiquant l'adresse
21 server.listen(port, hostname, () => {
22   console.log(`Server running at http://${hostname}:${port}/`);
23 });

```

Node JS – Initialisation d'un projet

- Les dépendances du serveur doivent être installées bien qu'aucune ne soit nécessaire pour notre cas.
- Dans tous les cas exécutez la commande **npm install**
- Démarrez ensuite le serveur avec la commande **node index.js**
- Le serveur tourne alors sur le port 3000
- Ouvrez PostMan
- Saisissez l'URL sur serveur Node.js : <http://localhost:3000>
- Exécutez la requête
- « Hello world » est retourné, bravo le serveur tourne !



Des questions ?