Node.js - Jour 1

Que fait-on à un dev quand il pleure ? On le console .... log

Introduction

Depuis plusieurs années, JavaScript a connu une évolution rapide. Aujourd'hui, ce langage peut être utilisé aussi bien côté **front-end** que côté **back-end** grâce à **Node.js**.

Node.js est un **environnement d'exécution JavaScript** côté serveur qui repose sur le moteur V8 de Google. Ainsi, il peut agir comme un serveur web, entre autres fonctionnalités. Node.js est capable de répondre à une multitude de besoins client/serveur.

Créez sur GitHub un répertoire nommé "**runtrack-nodeJS**". Dans ce répertoire, créez un dossier "**jour1**". Pour chaque job, créez un dossier "**jobXX**" ou **XX** est le numéro du job. Pour chaque job, un fichier "**index.js**" doit être créé contenant les instructions demandées. N'oubliez pas d'envoyer vos modifications dès qu'un job est avancé ou terminé et mettez des commentaires explicites lors de vos commits.

Installation

Pour commencer, installez votre environnement de développement. Vous devez installer Node.js et NPM. Assurez-vous d'avoir la dernière version de Node.js en exécutant la commande suivante dans votre terminal :

```
node -v
```

Job 1

Créez un script qui affiche le message suivant dans la console : "**Hello World!**". Exécutez votre programme à l'aide de la commande suivante :

```
node index.js
```

Tout fonctionne comme prévu ? Vous pouvez passer à la suite.

Job 2

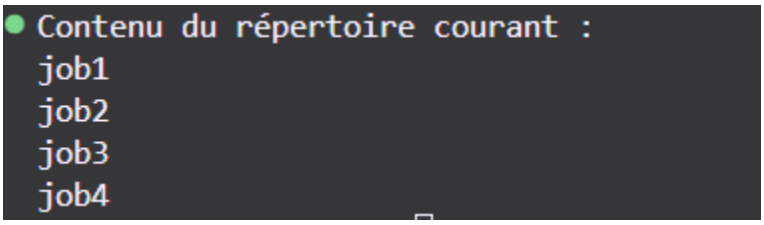
Créez deux variables, "**a**" et "**b**", ayant respectivement pour valeurs **8** et **12**. Additionnez ces deux variables et affichez le résultat dans le terminal. Exécutez votre script pour vous assurer que tout fonctionne comme prévu.

Job 3

Créez une variable nommée “**count**” et attribuez-lui la valeur **3**. Utilisez un **seul console.log()** pour afficher dans le terminal : “**La valeur de count est : 3**”. Exécutez votre script pour vous assurer que tout fonctionne comme prévu.

Job 4

À l’aide du module “**fs**”, affichez l’ensemble des dossiers présents dans le répertoire courant. Exécutez votre script pour vous assurer que tout fonctionne comme prévu.



Job 5

À l’aide d’un module “**path**” que vous aurez importé, réalisez les actions suivantes :

- ➔ Récupérez le nom du fichier.
- ➔ Récupérez l'extension du fichier.
- ➔ Récupérez le répertoire parent du fichier.

```
Nom du fichier: index.js
Extension du fichier: .js
Répertoire parent du fichier: C:\Bureau\sujets\job5
```

Job 6

Récupérez le contenu du fichier data.txt de façon **synchrone**, et afficher le contenu dans le terminal.

```
Contenu du fichier data.txt :
Explorez l'inconnu, créez l'inattendu, vivez l'extraordinaire.
```

Job 7

Récupérez le contenu du fichier data.txt de manière **asynchrone**, et affichez le contenu dans le terminal.

Job 8

Récupérez le contenu textuel du fichier data.txt et affichez dans le terminal **une lettre sur deux**.

```
Une Lettre sur deux du fichier data.txt :
Epoe 'non,cézliatnu ie 'xrodnie
```

Job 9

Écrire un programme qui modifie le contenu du fichier data.txt. Remplacez le contenu existant par : "**Je manipule les fichiers avec un module node !**".

Job 10

Écrire un script qui contient une constante nommée **URL** et qui a pour valeur : "**https://www.google.com&search=nodejs**".
À l'aide du module "**url**", faire les actions suivantes :

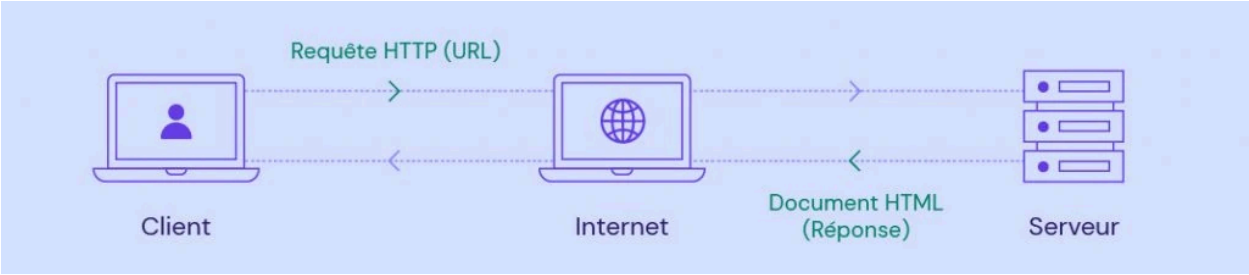
- ➔ Récupérer le protocole utilisé
- ➔ Récupérer le nom d'hôte
- ➔ Récupérer les paramètres de l'URL
- ➔ Reformater l'URL en une nouvelle URL valide en modifiant le nom hôte par "www.laplateforme.io"
- ➔ Ajouter à cette nouvelle URL un paramètre
- ➔ Afficher dans le terminal la nouvelle URL

```
Le protocole est  https:
Nouvelle URL :  https://www.laplateforme.io/?lang=fr
```

Serveur Web

Vous venez de manipuler différents modules Node.js. Comme vous avez pu le remarquer, rien de bien compliqué. Abordons dès à présent le module "***http***".

Ce module est un composant principal pour la création de serveurs web, la gestion des requêtes entrantes et l'envoi de requêtes HTTP vers d'autres serveurs.



Job 11

Créez votre premier serveur web Node.Js sur le port **8888** et affichez "**Hello World !**".
Bravo, vous avez créé votre premier serveur web avec Node.js.

Job 12

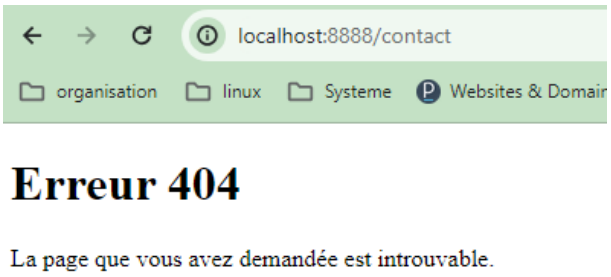
Créer un fichier "**index.html**" contenant les balises de bases et possédant dans son body un titre H1 de votre choix.
Créer un serveur web qui renvoie cette dernière. Lancer votre serveur et ouvrez un navigateur à l'adresse de votre localhost au bon port.

Job 13

Créer **deux fichiers HTML**, "**index.html**" qui possède les balises de base et un titre : "Page d'accueil" et "**about.html**" qui contient les balises de base ainsi qu'un titre : "Qui sommes-nous ?".
Créer un serveur web qui devra afficher la page "index.html" si l'URL "**/**" est appelée et "about.html" si "**/about**" est appelée.

Job 14

Reprenez le fichier créé ci-dessus et ajoutez une gestion d'erreur. Si une page autre que "**/**" ou "**/about**" est appelée, l'utilisateur doit être renvoyé sur la page "**error.html**". Cette dernière doit afficher un message d'erreur.



Compétences visées

- NodeJS

Rendu

Le projet est à rendre sur **https://github.com/prenom-nom/runtrack-nodeJs**.

Base de connaissances

- ➔ Installer Node.js
- ➔ Comment utiliser File System de node.js ?
- ➔ Module path
- ➔ Module HTTP
- ➔ Qu'est-ce qu'un serveur web ?
- ➔ Création d'un serveur web avec node.js