# JavaScript - Semaine 1 /6 – Séance 1/3

## Préambule

Le langage JavaScript peut s’utiliser de différentes façons dans une application Web. Pour ce cours de JavaScript, nous développerons des applications JavaScript côté serveur de type MPA (Multiple Pages Application). Plus précisément nous utiliserons Node.js avec le Framework Express.

## Objectifs de la semaine

* Prise en main de l'environnement de développement (Node.js / VScode)
* Savoir utiliser les éléments de base du langage JavaScript (condition, boucle, tableaux, …)
* Comprendre l'architecture Web Client-Serveur et son application avec Node.js
* Requête HTTP avec méthode GET
* Comprendre le système de modules dans Node.js
* Savoir installer des modules dans Node.js

**La majorité des exercices repartent de l’exercice précédent. Réalisez donc les exercices dans l’ordre. Pour conserver chaque exercice, sachez que vous pouvez facilement copier–coller une solution. Par exemple, quand vous avez fini l’exercice 2, vous pouvez recopier le répertoire de cette solution dans un nouveau répertoire nommé exercice 3.**

## Ex1 - Environnement de travail

Définissez et installez (si nécessaire) votre environnement de travail.

### OS

Vous pouvez travailler sous Windows ou Linux. A l’école, je vous conseille néanmoins Windows.

Node.js

Node.js doit être installé sur la machine. Sur les PC de l’école, l’installation a déjà été réalisée, il n’y a donc rien à faire.

Si vous désirez travailler sur votre PC portable, consultez les slides pour effectuer l’installation de Node.js.

Editeur

Nous travaillerons avec l’éditeur Visual Studio Code. Sur les PC de l’école, l’installation a déjà été réalisée, il n’y a donc rien à faire.

Si vous désirez travailler sur votre PC portable, consultez les slides pour effectuer l’installation de VsCode.

## Exercice 2 – Hello world !

Commençons par créer une première application en JavaScript avec Node.js et Express. Voici les étapes :

1. Créez un répertoire Exo1-HelloWorld
2. Ouvrez ce répertoire avec Visual Studio Code
3. Dans Visual Studio Code, affichez la fenêtre Terminal (Terminal -> New Terminal)
4. Créez une application Node.js (npm init à taper dans Terminal)
   1. package name -> laissez le choix par défaut
   2. version -> laissez le choix par défaut
   3. description -> description de l'application à remplir (Ex1 par exemple)
   4. entry point -> remplacez index.js par **app.js**
   5. test command -> laissez vide
   6. git repository -> laissez vide
   7. keywords -> laissez vide
   8. author -> mettez votre nom
   9. licence -> laissez ISC
5. Installez le framework Express ( npm install express à taper dans Terminal)
6. Créez un fichier nommé « app.js » à la racine du projet
7. Copiez le code suivant dans « app.js »

const express = require('express');

const app = express();

const port = 3000;

app.get('/', (req, res) => {

res.send('Hello World!');

});

app.listen(port, () => {

console.log(`Example app listening at http://localhost:${port}`);

});

1. Exécutez le code (node app.js à taper dans Terminal)
2. Suivez le lien via le ctrl-click (<http://localhost:3000>) et voyez le résultat
3. Arrêtez votre application via ctrl-c

## Exercice 3 – Date et Heure

Modifiez le premier exercice pour afficher la date et l’heure actuelle à Bruxelles.

**Quelques indications :**

1. Vous pouvez ajouter du code entre les accolades (donc au-dessus du res.send)
2. Vous pouvez définir une variable comme ceci :
   1. let mystring = « Bonjour »
3. Regardez la documentation suivante :
   1. <https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Global_Objects/Date> (la documentation MDN est très utile, n’hésitez pas à la consulter en tout temps)
   2. Internet peut également vous aider
4. Vous devez relancer (node app.js) votre serveur Node.js dès que vous faites une modification dans le code
5. Résultat attendu (Exemple ) :
   1. Nous sommes le 5/2/2022. Il est 9:30 à Bruxelles.

## Exercice 4 – Package.json et Nodemon

Intéressons-nous maintenant au fichier package.json. Faisons une première analyse du fichier package.json de l’exercice précédent.

1. Format JSON : le fichier est structuré en JSON. Renseignez-vous sur ce type de format sur Internet et donnez une simple description ici.

C’est un langage de transfert de données très léger. Il permet d’échanger des données.

1. Ce fichier contient tous les éléments que vous avez fourni à la commande “npm init”
2. Ce fichier contient toutes les dépendances des modules installés (Express ici)
3. Que signifie les chiffres derrière express et le tilde ?

C’est la version de la dépendance.

1. Où sont installées les dépendances (Express par exemple) ?

Dans le dossier node\_modules.

1. Express doit-il être installé une seule fois sur la machine ou par projet/application ?

Une fois par projet/application

1. C’est assez pénible de devoir relancer le serveur Node.js dès que l’on modifie le code. Nous allons donc améliorer ceci.
2. Installez nodemon dans les dépendances de développement :
   1. npm install nodemon --save-dev
3. Ajoutez un script dans le package.json (à la clef “scripts”) :
   1. "start": "nodemon app.js",
4. Maintenant vous pouvez lancer votre application comme ceci :
   1. npm start
5. Vous devez encore rafraîchir la page dans votre navigateur (F5 ou Actualiser)
6. Que s’est-il passé dans le fichier package.json ?

Une catégorie devDepencies est apparue avec nodemon