

Tutoriel NodeJS

1. Installation de NodeJS (sur une Ubuntu 14.04)

→ Commandes à faire en root :

```
curl -sL https://deb.nodesource.com/setup | sudo bash -
aptitude install python-software-properties python g++ make
aptitude update
aptitude install nodejs build-essential
```

Note : suivant cette méthode, *npm* est installé avec *nodejs*.

2. Création de la structure du projet

→ Commandes à faire en utilisateur :

```
mkdir StifCo
cd StifCo
mkdir public routes views
mkdir public/images public/javascripts
```

3. Création du fichier de configuration et installation des modules

→ Création du fichier `package.json` à la racine du projet :

```
{
  "name": "stifco",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "express": "*",
    "ejs": "*",
    "cookie-session": "*",
    "body-parser": "*"
  },
  "author": "Gilles Chamillard <gilles.chamillard@gmail.com>",
  "description": "Mission 4 - 2015 - BTS SIO"
}
```

Note : au lycée il faut prendre en compte le proxy (les deux premières lignes).

→ Installation des modules :

```
npm config set proxy http://proxy.vinci-melun.org:3128
npm config set https-proxy http://proxy.vinci-melun.org:3128
npm install
```

4. Création du fichier de l'application

→ Création du fichier `app.js` à la racine du projet :

```
// Modules
var express = require('express');
var routes = require('./routes');
var app = express();

// Configuration
app.set('views', __dirname + '/views');
app.set('view engine', 'ejs');
app.use(express.static(__dirname + '/public'));
```

```
// Routes et run
app.get('/', routes.index);
app.listen(8080);
```

6. Création du fichier contrôleur par défaut

→ Création du fichier `index.js` pour la route `/` à placer dans `/routes` :

```
// Contrôleur index

exports.index = function(req, res){
  res.render('index', { title: 'Express' })
};
```

7. Création du fichier de la vue par défaut

→ Création du fichier `index.ejs` associée au contrôleur à placer dans `/views` :

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <h1>Votre serveur NodeJS fonctionne avec <%= title %></h1>
  </body>
</html>
```

8. Création d'un fichier de styles pour la vue

→ Création du fichier `style.css` à placer dans `/public` :

```
body {
  padding: 50px;
  font: 14px "Lucida Grande", Helvetica, Arial, sans-serif;
}

a {
  color: #00B7FF;
}

h1 {
  color: #0000FF;
}
```

9. Test du serveur

→ Se positionner à la racine du projet :

```
node app.js
```

→ Voir le résultat dans un navigateur avec l'URL `http://localhost:8080`

Votre serveur NodeJS fonctionne avec Express

10. Installation du module MySQL et d'une base

→ Se positionner à la racine du projet et installer les deux modules :

```
npm install mysql
npm install express-myconnection
```

→ Créer une base nommée BD

→ Utiliser les commandes SQL (gestion d'albums de bandes dessinées) :

```
-- Base de données: `BD`

-- Structure de la table `auteur`

CREATE TABLE IF NOT EXISTS `auteur` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `nom` varchar(30) NOT NULL,
  `editeur` varchar(36) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=0 ;

-- Contenu de la table `auteur`

INSERT INTO `auteur` (`nom`, `editeur`) VALUES
('Dupa', 'Le Lombard'),
('Jean-Claude Mézières', 'Dargaud'),
('Moebius', 'Les Humanoïdes Associés');
```

11. Modification du fichier de configuration de l'application

→ L'installation des deux modules nécessite des changements dans le fichier `app.js`, avec l'utilisation des nouveaux modules, les paramètres de configuration pour l'accès à la base de données et une route supplémentaire (auteurs) :

```
// Modules
var express = require('express');
var mysql = require('mysql');
var routes = require('./routes');
var auteurs = require('./routes/auteurs');
var connection = require('express-myconnection');
var app = express();

// Configuration
app.set('views', __dirname + '/views');
app.set('view engine', 'ejs');
app.use(express.static(__dirname + '/public'));

app.use(
  connection(mysql, {
    host      : 'localhost',
    user      : 'root',
    password  : 'sio',
    database  : 'BD'
  }, 'pool')
);

// Routes
app.get('/', routes.index);
app.get('/auteurs', auteurs.list);

// Run
app.listen(8080);
```

12. Création du contrôleur et de la vue pour l'affichage de la table

→ Créez le contrôleur `auteur.js` à placer dans `/routes` :

```
// Contrôleur auteurs

exports.list = function(req, res){
  req.getConnection(function(err,connection){
    connection.query('SELECT * FROM auteur',function(err,rows) {
      if (err) throw err;
      res.render('auteurs',{page_title:"Auteurs - Node.js",data:rows});
    });
  });
};
```

Remarques :

On retrouve la pleinement la syntaxe de NodeJS à savoir l'exportation de la fonction `list` qui réalise les choses suivantes :

- obtention d'une connexion à la base de données
- réalisation d'une requête
- transmission à la vue associée d'un tableau pour l'affichage

Cette fonction sera exécutée avec l'URL `localhost:8080/auteurs` car on l'a dit dans le fichier `app.js` avec l'instruction `app.get('/auteurs', auteurs.list);`.

→ Créez ensuite la vue associée `auteur.ejs` à placer dans `/views` :

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <div class="page-data">
      <div class="data-btn">
        <button onClick="addUser();">+ Add</button>
      </div>
      <div class="data-table">
        <table border="1" cellpadding="7" cellspacing="7">
          <tr>
            <th width="50px">Id</th>
            <th>Nom</th>
            <th>Editeur</th>
          </tr>
          <% if(data.length){
            for(var i = 0;i < data.length;i++) { %>
            <tr>
              <td><%= (i+1) %></td>
              <td><%=data[i].nom%></td>
              <td><%=data[i].editeur%></td>
            </tr>
            <% }
          }else{ %>
            <tr>
              <td colspan="3">No user</td>
            </tr>
            <% } %>
          </table>
        </div>
```

```
</div>
</body>
</html>
```

→ Ajoutez un lien dans la vue par défaut `index.ejs` :

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <h1>Votre serveur NodeJS fonctionne avec <%= title %></h1>
    <a href="./auteurs" >Liste des auteurs de la base des bandes dessinées</a>
  </body>
</html>
```

→ Testez dans un navigateur :

<div>+ Add</div>		
Id	Nom	Editeur
1	Dupa	Le Lombard
2	Jean-Claude Mézières	Dargaud
3	Moebius	Les Humanoïdes Associés

Ce tutoriel est très fortement inspiré de celui-ci :

<http://teknosains.com/i/simple-crud-nodejs-mysql>

S'y reporter pour voir la suite des opérations CRUD...