Lab 3 : Introduction au framework express.js

Pour pouvoir réaliser les exercices de ce lab, on va commencer d'abord par :

1. Créer un repertoire principal /expressdemos qui va contenir tous les exercices suivants.

2. Dans ce nouveau répertoire, installer le framework express.js à partir de npm :

npm install express

Exercice 1 : Créer un serveur web simple avec express

1. Dans votre espace de travail, créer un répertoire /webserverwithexpress
2. Sous le nouveau répertoire, créer un fichier basicserver.js
3. Dans ce fichier saisir le code suivant :

```
var express = require('express'),
     http = require('http');

// Create an express application
var app = express()
            // register a middleware
            .use(function (req, res, next) {
                res.end('hello express!');
            });

// Register with http
http.createServer(app)
    .listen(3000);
```

4. Exécuter le fichier basicserver.js à l'aide de la commande node
5. Vérifier l'affichage du message : hello express ! dans le navigateur
6. Toujours sous le répertoire /webserverwithexpress, ajouter un nouveau fichier simplerserver.js
7. Dans ce nouveau fichier, saisir le code suivant :

```
var express = require('express');

express()
    .use(function (req, res, next) {
        res.end('hello express!');
    })
    .listen(3000);
```

8. Lancer le serveur et vérifier que vous obtenez le même résultat que pour l'exemple précédent.

Exercice 2 : Servir des fichiers statiques avec express

1. Dans le répertoire principal, installer le module serve-static à partir de npm.

2. Créer un nouveau répertoire /servestaticpages
3. Dans le nouveau répertoire créer un répertoire /public contenant un fichier index.html.
4. Sous /servestaticpages, ajouter un fichier staticserver.js contenant le code suivant :

```
var express = require('express');
var serveStatic = require('serve-static');

var app = express()
    .use(serveStatic(__dirname + '/public'))
    .listen(3000);
```

ou plus simplement :

```
var express = require('express');

var app = express()
    .use(express.static(__dirname + '/public'))
    .listen(3000);
```

5. Sous le répertoire /public, ajouter un fichier home.html puis essayer d'accèder à l'URL : localhost :3000/home.html
Que remarquez-vous ?
6. Essayer d'ajouter une image et une feuille de style à votre fichier home.html. Que faut-il ajouter pour que ces nouveaux fichiers soient servis avec le fichier html ?

Exercice 3 : Accepter des données JSON ou des données de formulaire à l'aide de body-parser

1. Dans le répertoire principal, installer le module body-parser à partir de npm.
2. Créer un nouveau répertoire /dataparser
3. Sous le nouveau répertoire, ajouter un fichier jsonparser.js contenant le code suivant :

```
var express = require('express');
var bodyParser = require('body-parser');

var app = express()
    .use(bodyParser())
    .use(function (req, res) {
        if (req.body.foo) {
            res.end('Body parsed! Value of foo: ' + req.body.foo);
        }
        else {
            res.end('Body does not have foo!');
        }
    })
    .use(function (err, req, res, next) {
        res.end('Invalid body!');
    })
    .listen(3000);
```

4. Lancer le serveur puis Installer l'application postman (https://www.getpostman.com/) pour la réponse du serveur lorsque vous envoyez un objet JSON {"foo":123}
5. Tester la réponse du serveur lorsqu'on lui envoi des données de formulaire contant la valeur foo="123"

Exercice 5 : Créer des routes avec express
1. Sous le répertoire principal, ajouter un répertoire /routing
2. Sous /routing, ajouter un fichier approuting.js contenant le code suivant

```
var express = require('express');

var app = express();
app.all('/', function (req, res, next) {
    res.write('all\n');
    next();
});
app.get('/', function (req, res, next) {
    res.end('get');
});
app.put('/', function (req, res, next) {
    res.end('put');
});
app.post('/', function (req, res, next) {
    res.end('post');
});
app.delete('/', function (req, res, next) {
    res.end('delete');
});
app.listen(3000);
```

3. A l'aide de postman, tester les résultats des routes définies ci-dessus.