TDM06 de Technologies Web: Node.js (1)

ASI4 - INSA Rouen

CORRECTION

1 Serveur Node.js

— Réalisez un premier serveur qui affiche une page html générée dynamiquement. Cette page HTML contiendra un formulaire avec différents objets (champs texte, boutons, etc.).

Correction

```
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/html');
        res.end(`<!DOCTYPE html>
                  <html>
                    <body>
                      <form action="" method="GET">
                        <label>Name : </label><input type="text" name="name" size="30">
                        <input type="submit" value="Submit">
                      </form>
                    </body>
                  </html>`);
});

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

— Créer un script qui affiche la liste des paramètres envoyés en GET.

Correction

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

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
        let reponse = 'Liste des paramètre de la requête GET:\n';
        res.statusCode = 200;
        res.setHeader('Content-Type', 'text/plain; charset=utf-8');
        const queryObject = url.parse(req.url, true).query;
        for(element in queryObject){
          reponse += `${element} : ${queryObject[element]}\n`;
          }
        res.end(reponse);
        });

server.listen(port, hostname, () => {
  console.log(`Server listening on port ${port}`);
});
```

— Installez Express afin de gérer les ressources statiques dans un répertoire spécifique (e.g. 'public').

Correction

```
const express = require('express');
const app = express();
app.use(express.static('public'));

const port = 3000;

app.listen(port, () => {
  console.log(`Express app listening on port ${port}`);
});
```

— Modifiez votre serveur pour que :
  — un point d'entrée sans paramètre redirige vers un formulaire statique ;
  — le formulaire soit traité, en GET, par le script conçu précédemment.

Correction

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

const express = require('express');
const app = express();
app.use(express.static('public'));

const url = require('url');

const hostname = '127.0.0.1';
const portHttp = 3030;
const portExpress = 3000;

const server = http.createServer((req, res) => {
  if(url.parse(req.url, false).query==null){
    res.writeHead(303, { location: "http://localhost:3000/index.html" });
    res.end();
  } else {
    const queryObject = url.parse(req.url, true).query;
    res.setHeader('Content-Type', 'text/plain; charset=utf-8');
    let reponse = 'Liste des paramètre de la requête GET:\n';
    res.statusCode = 200;
    for(element in queryObject){
      reponse += `${element} : ${queryObject[element]}\n`;
      }
    res.end(reponse);
  }
  });

server.listen(portHttp, hostname, () => {
  console.log(`Server listening on port ${portHttp}`);
});
app.listen(portExpress, () => {
  console.log(`Express app listening on port ${portExpress}`);
});
```

2 Compteur Node.js

L'objectif de l'exercice est de développer le code nécessaire à la création d'un compteur sur une page web. Ce compteur s'incrémentera à chaque chargement de la page (i.e. depuis plusieurs postes différents) dans une première version. Dans une seconde version, vous ferez en sorte que le redémarrage du serveur ne réinitialise pas le compteur.

Correction

1ère version

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

const hostname = '127.0.0.1';
const port = 3000;

let hits = 0;

const server = http.createServer((req, res) => {
    res.setHeader('Content-Type', 'text/plain; charset=utf-8');
    res.statusCode = 200;
    hits++;
    res.end(`Nombre de connexions: ${hits}`);
  });

server.listen(port, hostname, () => {
  console.log(`Server listening on port ${port}`);
});
```

2nde version

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

const hostname = '127.0.0.1';
const port = 3000;

function lire_hits() {
    try {
      return(parseInt(fs.readFileSync('hits.txt', 'utf8')));
    } catch (err) {
      console.error(err);
    }
```

```
}

function ecrire_hits(hits) {
    fs.writeFile('hits.txt', hits.toString(), function (err) {
      if (err)
        console.log('Problème écriture de fichier');
    });
}

const server = http.createServer((req, res) => {
    res.setHeader('Content-Type', 'text/plain; charset=utf-8');
    res.statusCode = 200;
    let hits = lire_hits();
    hits++;
    ecrire_hits(hits);
    res.end(`Nombre de connexions: ${hits}`);
  });

server.listen(port, hostname, () => {
  console.log(`Server listening on port ${port}`);
});
```

Remarques

1. Continuez à vérifier vos pages (`http://validator.w3.org/` ou `http://www.xmlvalidation.com/`).
2. À l'issu de la séance, vous aurez accès à la correction de ce TDM au format PDF.
3. **Déposez votre compte-rendu sur moodle sous la forme d'un fichier PDF nommé `TDM06-login.pdf`, chez chacune des 2 personnes du binôme.**